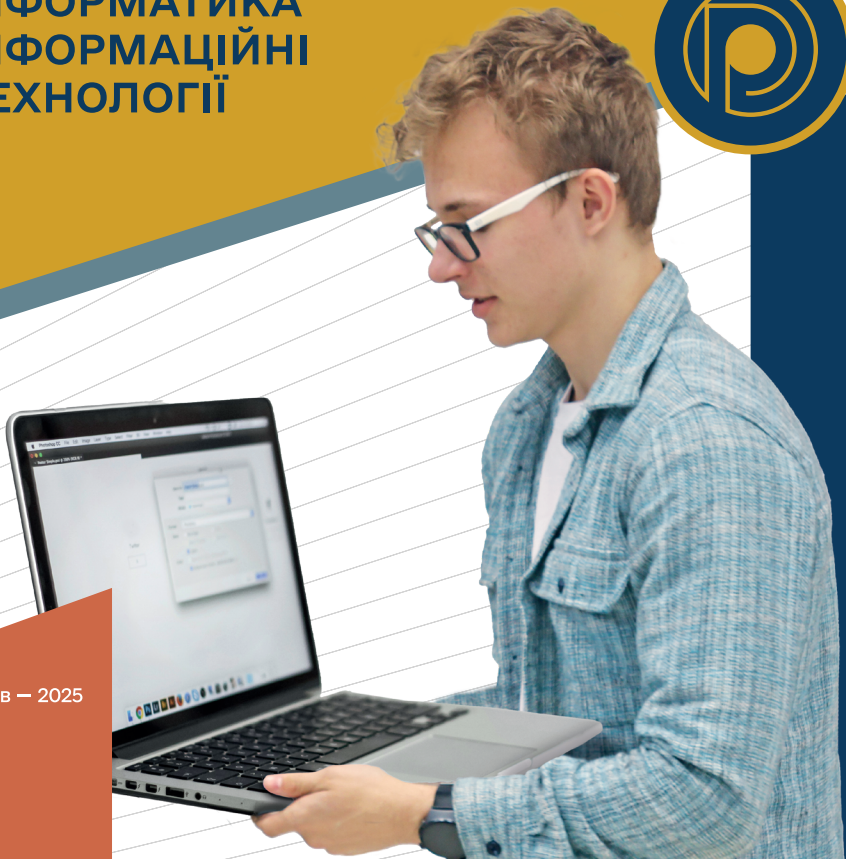
Завдання

IV етапу Всеукраїнських
учнівських олімпіад
з навчальних предметів

2024/2025 навчальний рік

ІНФОРМАТИКА
ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ



Київ — 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
НАЦІОНАЛЬНИЙ ЦЕНТР «МАЛА АКАДЕМІЯ НАУК УКРАЇНИ»

Завдання

IV етапу Всеукраїнських
учнівських олімпіад
з навчальних предметів

2024/2025 навчальний рік

ІНФОРМАТИКА

**ІНФОРМАЦІЙНІ
ТЕХНОЛОГІЇ**

Практикум

Київ
Національний центр
«Мала академія наук України»
2025

УДК 373
З-13

Автори:

І. Г. Бондік, *Ю. В. Горошко*, д. пед. наук, проф., *Г. Ю. Громко*, *К. І. Денисов*, *О. А. Журибеда*, *М. М. Кузічев*, *В. А. Кушнір*, *Т. Г. Ніколаєв*, *М. Р. Паламарюк*, *О. В. Шаравара*

Відповідальна за випуск:
А. І. Грітчина, канд. пед. наук

Упорядники:
О. М. Косьмій, канд. політ. наук, доц., *О. Б. Липовецька,* канд. мед. наук, *О. В. Роговець,* канд. пед. наук.

Рекомендовано для використання в освітньому процесі рішенням науково-методичної ради Національного центру «Мала академія наук України» (протокол № 2 від 16.06.2025)

З-13 **Завдання** IV етапу Всеукраїнських учнівських олімпіад з навчальних предметів. 2024/2025 навчальний рік. Інформатика, інформаційні технології : практикум / І. Г. Бондік, Ю. В. Горошко, Г. Ю. Громко та ін. ; відп. за вип. А. І. Грітчина ; упоряд.: О. М. Косьмій, О. Б. Липовецька, О. В. Роговець. – Київ : Національний центр «Мала академія наук України», 2025. – 88 с.
ISBN 978-617-7945-98-6

Практикум, створений для ефективної підготовки учнів до участі в олімпіадах, містить завдання IV етапу Всеукраїнської учнівської олімпіади з інформатики та інформаційних технологій у 2024/2025 навчальному році.

Завдання з інформатики спрямовані на перевірку умінь розв'язувати алгоритмічні задачі за допомогою комп'ютера, а з інформаційних технологій – на використання пакета MS Office (Access, Excel Word, PowerPoint).

Видання призначене для учнів 10–11 класів і педагогічних працівників, які їх супроводжують під час підготовки до участі в інтелектуальних змаганнях.

УДК 373

ISBN 978-617-7945-98-6

ЗМІСТ

ВСТУП

Інформатика та інформаційні технології є рушіями цифрової трансформації суспільства й розвитку економіки знань. Формування у молодого покоління цифрової грамотності, алгоритмічного мислення, вміння аналізувати, моделювати та розв'язувати складні задачі — це стратегічне завдання освіти, спрямоване на підготовку компетентних, творчих фахівців з інноваційним мисленням.

З огляду на зазначене організація і проведення олімпіад спрямовані на підтримку талановитої молоді, формування навичок оброблення даних, їх аналізу і візуалізації, розв'язання алгоритмічних завдань за допомогою мов програмування.

У збірнику представлені завдання, що були запропоновані учасникам IV етапу Всеукраїнських учнівських олімпіад з інформатики та інформаційних технологій у 2024/2025 навчальному році.

Олімпіадні завдання з інформатики передбачають розв'язування алгоритмічних задач за допомогою комп'ютера, а з інформаційних технологій — використання пакета MS Office (Access, Excel Word, PowerPoint). Учасники демонструють вміння знаходити раціональні й нестандартні рішення, а також розбивати складні задачі на прості кроки, складати плани дій для досягнення мети (алгоритми), аналізувати й виявляти помилки у послідовності дій, бачити причинно-наслідкові зв'язки. Ці навички є фундаментальними не лише для програмування, а й для виконання професійних і будь-яких побутових завдань.

Запропонований практикум може бути використаний як для подальшої підготовки до інтелектуальних змагань, так і в освітньому процесі задля поглиблення знань учнів та їх зацікавлення комп'ютерними науками.

Збірник стане у пригоді учням, які готуються до участі в олімпіадах, вчителям, викладачам, педагогам, які супроводжують цей процес.

Анна Грітчина,
заступник директора
з методичної роботи НЦ «МАНУ»,
кандидат педагогічних наук

ІНФОРМАТИКА

Умови задач

I тур

1. Хлопці та дівчата

Існує n типів хлопців та $2 \times n$ дівчат. Типи хлопців пронумеровані цілими числами від 1 до n, а дівчата пронумеровані цілими числами від 1 до $2 \times n$.

Хлопців i-го типу є c_i, кожному з них подобаються дівчата з номерами a_i та b_i.

Знайдіть розмір найбільшої множини хлопців, такої, що для кожної пари хлопців з цієї множини існує хоча б одна дівчина, яка подобається їм обом.

У цій задачі кожен тест містить кілька наборів вхідних даних.

Вам потрібно розв'язати задачу окремо для кожного такого набору.

Формат вхідних даних

У першому рядку задано одне ціле число T ($1 \le T \le 500$) — кількість наборів вхідних даних.

Далі задано опис наборів вхідних даних.

У першому рядку кожного набору вхідних даних задано одне ціле число n ($1 \le n \le 7 \times 10^5$).

У наступних n рядках задано по три цілі числа a_i, b_i, c_i ($1 \le \le a_i < b_i \le 2 \le n$, $1 \le c_i \le \times 10^9$) — параметри відповідного типу хлопців.

Гарантується, що $a_i \ne a_j$ або $b_i \ne b_j$ для будь-яких $1 \le i < j \le n$.

Гарантується, що сума n у всіх наборах вхідних даних одного тесту не перевищує 7×10^5.

Формат вихідних даних

Для кожного набору вхідних даних виведіть в окремому рядку одне ціле число — розмір найбільшої множини хлопців, такої, що для кожної пари хлопців з цієї множини існує хоча б одна дівчина, яка подобається їм обом.

Система оцінювання

Нехай S — сума n у всіх наборах вхідних даних одного тесту, а K — сума всіх c_i у всіх наборах вхідних даних одного тесту.

5 балів: $n \leq 5$;

11 балів: $S \leq 100$;

7 балів: кожна дівчина подобається хлопцям не більше ніж двох типів;

10 балів: $S \leq 3000$;

23 бали: $S \leq 3 \times 10^5$;

19 балів: $K \leq 10^7$;

25 балів: без додаткових обмежень.

Приклад

standard input	standard output
3 2 1 2 3 3 4 5 5 1 2 1 1 3 4 4 5 2 3 4 2 1 4 3 4 1 2 3 2 3 4 3 5 4 1 3 2	5 9 10

2. Розбиття на три

По колу розставлені n невід'ємних цілих чисел a_1, a_2, ..., a_n. Сусідніми в порядку по колу є числа a_1 та a_2, a_2 та a_3, ..., a_{n-1} та a_n, a_n та a_1.

Розбийте ці числа на три непорожні групи так, щоб кожне число належало рівно одній групі, числа з однієї групи були розташовані послідовно по колу, а різниця між максимальною та мінімальною сумами чисел у групах була мінімально можливою.

Формат вхідних даних

У першому рядку задано одне ціле число n ($3 \le n \le 10^6$) — кількість розставлених чисел.

У другому рядку задано n невід'ємних цілих чисел $a_1, a_2, ..., a_n$ ($0 \le \le a_i \le 10^9$) — розставлені по колу числа.

Формат вихідних даних

У першому рядку виведіть одне ціле число — різницю між максимальною та мінімальною сумами чисел у групах в оптимальному розбитті.

У другому рядку виведіть три цілі числа x, y, z ($1 \le x < y < z \le n$) — такі номери, що оптимальне розбиття чисел на три групи має вигляд: $[a_x, a_{x+1}, ..., a_{y-1}]$, $[a_y, a_{y+1}, ..., a_{z-1}]$, $[a_z, a_{z+1}, ..., a_{n-1}, a_n, a_1, a_2, ..., a_{x-1}]$.

Якщо існує кілька правильних відповідей, то дозволяється вивести будь-яку з них.

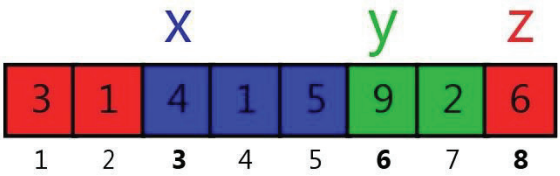
Система оцінювання

2 бали: $n = 3$;
4 бали: $a_i \le 1$ для $1 \le i \le n$;
13 балів: існує розбиття, де шукана різниця дорівнює 0;
8 балів: $n \le 100$;
9 балів: $n \le 2000$;
13 балів: $n \le 5000$;
28 балів: $n \le 10^5$;
23 бали: без додаткових обмежень.

Приклади

standard input	standard output
4 1 2 3 4	1 1 3 4
7 1 6 1 0 5 3 2	0 2 3 6
8 3 1 4 1 5 9 2 6	1 3 6 8

Примітка. У третьому прикладі оптимальне розбиття виглядає наступним чином:



У такому випадку суми в групах дорівнюють 10, 11 і 10.

3. Випуклий масив

Задано масив цілих чисел a довжини n.

Визначте, чи існує перестановка його елементів b така, що для кожного $2 \le i \le n - 1$ виконується:

$$b_{i-1} + b_{i+1} \ge 2 \times b_i.$$

У цій задачі кожен тест містить кілька наборів вхідних даних. Вам потрібно розв'язати задачу *окремо* для кожного такого набору.

Формат вхідних даних

У першому рядку задано одне ціле число T ($1 \le T \le 10^5$) — кількість наборів вхідних даних.

Далі задано опис наборів вхідних даних.

У першому рядку кожного набору вхідних даних задано одне ціле число n ($3 \le n \le 3 \times 10^5$) — довжина масиву a.

У другому рядку кожного набору вхідних даних задано n цілих чисел $a_1, a_2, ..., a_n$ ($0 \le a_i \le 10^9$) — елементи масиву a.

Гарантується, що сума n у всіх наборах вхідних даних одного тесту не перевищує 3×10^5.

Формат вихідних даних

Для кожного набору вхідних даних виведіть в окремому рядку «YES», якщо шукана перестановка існує, і «NO» — в протилежному випадку.

Система оцінювання

Нехай S — сума n у всіх наборах вхідних даних одного тесту.
3 бали: $n = 4$;
4 бали: $T = 1$, $n \leq 7$;
7 балів: $T = 1$, $n \leq 15$;
5 балів: якщо якась шукана перестановка існує, то існує така шукана перестановка, для якої виконується: $b_1 \geq b_2$ та $b_2 \leq b_3$;
17 балів: $S \leq 50$;
10 балів: $S \leq 400$;
13 балів: $S \leq 2000$;
9 балів: $S \leq 8000$;
18 балів: $a_i \leq 106$ для $1 \leq i \leq n$;
14 балів: без додаткових обмежень.

Приклад

standard input	standard output
10 4 0 3 4 6 4 5 4 1 4 8 1 4 4 8 6 10 10 4 7 2 1 5 1 9 4 6 6 7 1 6 10 2 3 7 6 6 10 2 5 3 8 4 9 9 1 5 4 8 4 3 4 7 1 2 1 6 4 2 9 7 3 9 7 5 9 10 10	YES NO NO YES YES NO YES YES YES NO

Примітка. У першому наборі вхідних даних першого прикладу перестановками масиву [0, 3, 4, 6], які задовольняють описану умову, є [4, 0, 3, 6] та [6, 3, 0, 4].

4. Проста підпослідовність

Назвемо масив цілих чисел $d_1, d_2, \ldots, d_m$ *хорошим*, якщо його довжина становить 0, або для будь-якого $1 \le i \le m$ обидва значення $\sum_{j=1}^{i} d_j$ та $\sum_{j=i}^{m} d_j$ є невід'ємними. Тут $\sum_{j=l}^{r} d_j$ позначає: $d_l + d_{l+1} + \cdots + d_r$.

Визначимо *красу* масиву як довжину найдовшої його *хорошої* підпослідовності*. Задано масив a довжини n, що складається з чисел −1 та 1.

Необхідно виконати q запитів. Запити бувають двох типів:

1) замінити елемент a_p на $-a_p$, де p — параметр запиту;

2) знайти *красу* масиву, що складається з елементів $[a_l, a_{l+1}, \ldots, a_r]$, де (l, r) — параметри запиту.

Формат вхідних даних

У першому рядку задано два цілі числа n, q ($1 \le n, q \le 5 \times 10^5$) — довжина масиву a та кількість запитів відповідно.

У другому рядку задано n цілих чисел $a_1, a_2, \ldots, a_n$ ($a_i \in \{-1, 1\}$) — елементи масиву a.

У наступних q рядках задано опис запитів. Перше з чисел $type_i$ ($1 \le type_i \le 2$) позначає тип запиту. Запити першого типу задані у форматі «1 p» ($1 \le p \le n$), а запити другого типу задані у форматі «2 l r» ($1 \le l \le r \le n$).

Формат вихідних даних

Для кожного запиту другого типу в окремому рядку виведіть одне ціле число — *красу* відповідного масиву.

Система оцінювання

2 бали: $a_i = (-1)^{i+1}$ для $1 \le i \le n$, а також немає запитів першого типу;
7 балів: $n \le 16$, а також немає запитів першого типу;
21 бал: $n, q \le 100$;
20 балів: $n, q \le 3000$;
27 балів: $n, q \le 2 \times 10^5$, а також немає запитів першого типу;
14 балів: $n, q \le 2 \times 10^5$;
9 балів: без додаткових обмежень.

Приклади

standard input	standard output
5 4 1 1 1 -1 1 2 1 5 1 3 2 1 4 2 2 5	5 2 3
4 4 1 1 1 -1 2 1 2 2 2 4 2 3 3 2 3 4	2 2 1 1

Примітка

* Масив *c* називається підпослідовністю масиву *b*, якщо можна видалити певну кількість елементів з масиву *b* (можливо, нульову) так, щоб утворився масив *c*. Порожній масив є підпослідовністю будь-якого масиву.

II тур

5. Манхетенське парування

Для пари точок на декартовій площині (x_1, y_1) та (x_2, y_2) визначимо манхетенську відстань між ними як $|x_1 - x_2| + |y_1 - y_2|$.

Наприклад, для пари точок $(4, 1)$ та $(2, 7)$ манхетенська відстань між ними дорівнює: $|4 - 2| + |1 - 7| = 2 + 6 = 8$.

Задано $2 \times n$ точок на декартовій площині, координати яких є цілими числами. Всі y-координати заданих точок дорівнюють 0 або 1.

Розбийте задані точки на n пар так, щоб кожна з цих точок належала тільки одній парі, а максимальна манхетенська відстань між точками однієї пари була мінімально можливою.

Формат вхідних даних

У першому рядку задано одне ціле число n ($1 \le n \le 10^5$).

У наступних $2 \times n$ рядках задано по два цілі числа x_i, y_i ($0 \le x_i \le 10^9$, $0 \le y_i \le 1$) — координати відповідної точки.

Формат вихідних даних

Виведіть одне ціле число — максимальну манхетенську відстань між точками однієї пари в оптимальному розбитті.

Система оцінювання

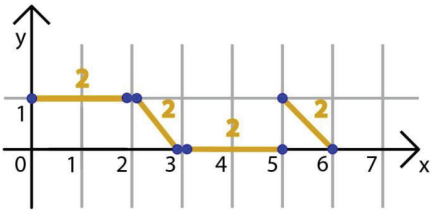
2 бали: $n = 1$;
3 бали: $x_i = 0$ для $1 \le i \le 2 \times n$;
4 бали: $n \le 4$;
11 балів: $n \le 10$;
14 балів: $y_i = 0$ для $1 \le i \le 2 \times n$;
10 балів: $x_i \ne x_j$ для $1 \le i < j \le 2 \times n$;
29 балів: $n \le 1000$;
27 балів: без додаткових обмежень.

Приклади

standard input	standard output
1 3 1 1 0	3
3 18 0 3 0 1 0 10 0 8 0 14 0	4
4 3 0 0 1 5 0 2 1 6 0 3 0 5 1 2 1	2

Примітка. У другому прикладі розбиття на пари [(18, 0), (14, 0)], [(3, 0), (1, 0)] та [(8, 0), (10, 0)] є єдиним оптимальним розбиттям. Манхетенські відстані між точками однієї пари в такому розбитті дорівнюють 4, 2 та 2 відповідно.

У третьому прикладі (див. рис.) розбиття на пари [(0, 1), (2, 1)], [(2, 1), (3, 0)], [(3, 0), (5, 0)] та [(5, 1), (6, 0)] є оптимальним розбиттям. Усі манхетенські відстані між точками однієї пари в такому розбитті дорівнюють 2.



6. Подарунок для Антона

Антон хоче отримати в подарунок прямокутну таблицю $n \times m$ з чисел 0, 1, 2, 3 або 4.

Антон буде щасливий, якщо поруч з кожним «0» не буде жодного іншого «0», поруч із кожною «1» буде рівно одна інша «1», поруч із кожною «2» буде рівно дві інших «2», поруч із кожною «3» буде рівно три інших «3» , та поруч із кожною «4» має бути чотири інших «4» (тобто всі сусіди «4» мають бути теж «4»).

Одна клітинка знаходиться поруч із іншою, якщо вони мають спільну сторону.

Вам потрібно придумати таблицю, яку можна подарувати Антону, щоб він був щасливий.

Нижче наведено приклад таблиці, яка зробить Антона щасливим, якщо $n = 4$, $m = 6$.

1	1	2	2	2	1
0	2	2	0	2	1
1	2	0	2	2	0
1	2	2	2	1	1

Формат вхідних даних

Єдиний рядок містить два цілі числа n, m ($1 \le n, m \le 200$) — розміри таблиці. Можна показати, що відповідь завжди існує.

Формат вихідних даних

Виведіть таблицю з n рядків та m стовпців — подарунок для Антона.

Система оцінювання

10 балів: $n = 1$;
10 балів: $n = 2$;
10 балів: $n = 3$;
10 балів: $n = m = 4$;
10 балів: $n = m = 30$;
5 балів: $n = 30$, $m = 31$;
5 балів: $n = 30$, $m = 32$;
10 балів: $n = m = 31$;

5 балів: $n = 31, m = 32$;
10 балів: $n = m = 32$;
15 балів: без додаткових обмежень.

Приклад

standard input	standard output
4 6	1 1 2 2 2 1 0 2 2 0 2 1 1 2 0 2 2 0 1 2 2 2 1 1

7. Розворот ABC

Задано рядок s, що складається з символів A, B та C.

За одну операцію дозволяється вибрати два сусідніх елементи рядка $s_i s_{i+1}$, які в такій послідовності дорівнюють AB, BC або CA, і поміняти їх місцями.

Знайдіть найбільшу кількість послідовних операцій, які можна виконати з рядком s.

У цій задачі кожен тест містить кілька наборів вхідних даних. Вам потрібно розв'язати задачу окремо для кожного такого набору.

Формат вхідних даних

У першому рядку задано одне ціле число T ($1 \le T \le 10^5$) — кількість наборів вхідних даних.

Далі задано опис наборів вхідних даних.

У першому рядку кожного набору вхідних даних задано одне ціле число n ($1 \le n \le 10^6$) — довжина рядка s.

У другому рядку кожного набору вхідних даних задано рядок s довжини n, що складається з символів A, B та C.

Гарантується, що сума n у всіх наборах вхідних даних одного тесту не перевищує 10^6.

Формат вихідних даних

Для кожного набору вхідних даних виведіть в окремому рядку одне ціле число — найбільшу кількість послідовних операцій, які можна виконати з рядком s.

Система оцінювання

Нехай K — сума n у всіх наборах вхідних даних одного тесту.

2 бали: відповідь становить 0 або 1;

3 бали: $n \le 3$;

5 балів: $si \ne C$ для $1 \le i \le n$;

5 балів: s має вигляд AA ... AABB ... BBCC ... CC (тобто $x \times A + y \times \times B + z \times C$ для певних додатних x, y, z);

9 балів: $sisi+1 \ne CA$ для $1 \le i < n$;

10 балів: $T = 1, n \le 12$;

8 балів: $n \le 12$;

28 балів: $K \le 2000$;

30 балів: без додаткових обмежень.

Приклад

standard input	standard output
2 5 ABCCA 19 CCAABBBABBAAABBCCAA	3 28

Примітка. У першому наборі вхідних даних першого прикладу можна виконати не більше ніж 3 послідовні операції з рядком ABCCA. Розглянемо один із прикладів того, як 3 послідовні операції можна виконати:

[ABCCA → BACCA, BACCA → BACAC, BACAC → BAACC].

8. Проста задача

Назвемо непорожню послідовність додатних цілих чисел *дивною*, якщо сума її елементів є простим числом.

Задано масив a довжини n, що складається з простих чисел. Також задано ціле число k. Розбийте масив a на k непорожніх підпослідовностей так, щоб кожен елемент масиву a належав тільки одній з них, а кількість *дивних* підпослідовностей серед утворених була мінімально можливою.

Примітка. Послідовність c називається підпослідовністю масиву b, якщо можна видалити певну кількість елементів з масиву b (можливо, нульову) так, щоб утворилась послідовність c.

У цій задачі кожен тест містить кілька наборів вхідних даних.

Вам потрібно розв'язати задачу окремо для кожного такого набору.

Зауважте, що в цій задачі немає блоку «без додаткових обмежень».

Формат вхідних даних

У першому рядку задано одне ціле число T ($1 \le T \le 10^5$) — кількість наборів вхідних даних.

Далі задано опис наборів вхідних даних.

У першому рядку кожного набору вхідних даних задано два цілі числа n, k ($1 \le k \le n \le 10^5$) — довжина масиву a та кількість підпослідовностей, на які його потрібно розбити.

У другому рядку кожного набору вхідних даних задано n простих чисел $a_1, a_2, ..., a_n$: ($2 \le a_i \le 10^5$, $a_i \le a_{i+1}$) — елементи масиву a.

Гарантується, що сума n в усіх наборах вхідних даних одного тесту не перевищує 10^5.

Формат вихідних даних

Для кожного набору вхідних даних виведіть оптимальне розбиття у наступному форматі:

- у першому рядку виведіть одне ціле число m — кількість *дивних* підпослідовностей серед утворених;
- у кожному з наступних k рядків виведіть цілі числа s_i та $b_1, b_2, ..., b_{s_i}$ ($1 \le s_i \le n$) — кількість елементів у відповідній підпослідовності розбиття та самі її елементи.

Підпослідовності та їхні елементи дозволяється виводити в будь-якому порядку. Якщо існує кілька правильних відповідей, дозволяється вивести будь-яку з них.

Система оцінювання

2 бали: $T \le 20$, $k = 1$;
5 балів: $n \le 4$, $a_i \le 10^4$ для $1 \le i \le n$;
8 балів: $T \le 20$, $n \le 10$, $a_i \le 10^4$ для $1 \le i \le n$;
4 бали: n — парне, $a_i > 2$ для $1 \le i \le n$;
18 балів: n — непарне, $a_i > 2$ для $1 \le i \le n$;
10 балів: $2 \times k \ge n + 1$;
29 балів: n — парне;
24 бали: n — непарне.

Приклад

standard input	standard output
4 3 1 5 5 13 4 2 2 3 57 5 3 3 3 55 13 6 5 2 2 23 33	1 3 13 5 5 0 2 27 2 35 1 1 13 2 33 2 55 4 1 2 1 2 1 2 1 3 2 33

Розв'язання задач

I тур

1. Хлопці та дівчата

Блок 1

Якщо в множині вже є хоча б один хлопець певного типу x, то всі інші хлопці цього ж типу також мають бути присутні. Отже, ми можемо перебирати всі підмножини типів хлопців, перевіряти, чи вони задовольняють умову задачі, та вибирати найкращий варіант.

Блок 2

Розглянемо задачу як граф, де вершини відповідають дівчатам, а хлопці є ребрами, що з'єднують ці вершини. Нам потрібно, щоб усі ребра в множині мали хоча б одну спільну вершину. Такі структури можуть бути лише двох типів:

- зірка — всі ребра виходять із однієї вершини;
- трикутник — три вершини, кожна з яких з'єднана з двома іншими.

Щоб знайти відповідь для зірки, достатньо визначити її центр x і підрахувати суму всіх ребер, що виходять із цієї вершини.

Для трикутника потрібно перебрати всі ребра (u, v) і знайти вершину, разом із якою вони утворюють трикутник.

Блок 3

Оскільки кожна вершина може мати не більше двох вихідних ребер, варто перебрати всі вершини, що мають рівно два сусіди, і перевірити, чи ці сусіди також з'єднані між собою. Якщо так, то знайдено трикутник.

Блок 4

У цій групі достатньо для кожної вершини x перебрати всі пари її сусідів (u, v). Потім перевіряємо, чи існує ребро між вершинами u і v. Це працює за складністю $O(n^2)$, оскільки кожна вершина може мати до n сусідів.

Блок 5

Щоб швидко знайти всі трикутники, можна розділити вершини на важкі (з більш ніж $\sqrt{n}$ сусідами) і легкі (з меншою кількістю сусідів).

Спочатку рахуємо всі трикутники, що містять хоча б одне легке ребро. Це можна зробити так само, як у попередньому підході, і це займе

$\mathcal{O}(n\sqrt{n})$ часу. Потім знаходимо трикутники, що складаються тільки з важких вершин. Для цього видаляємо всі легкі вершини та повторюємо підрахунок. Це також виконується за $\mathcal{O}(n\sqrt{n})$.

Загальна складність алгоритму — $\mathcal{O}(n\sqrt{n})$.

Блоки 6–7

Щоб знайти найкращий трикутник, потрібно для кожної вершини x розглянути трикутник, утворений двома найбільшими ребрами, що виходять з x (якщо такий існує). Таких трикутників є $O(n)$, тому отримаємо розв'язок за $O(n\log n)$ або $O(n)$ залежно від способу пошуку третього ребра таких трикутників.

Доведемо, що якщо оптимальна відповідь є трикутником, то ми її розглянемо. Дійсно, нехай оптимальна відповідь є трикутником зі сторонами, що мають ваги l_1, l_2, l_3, $l_1 \leq l_2 \leq l_3$. Нехай з вершини, що знаходиться між ребрами l_2 та l_2 (позначимо її v), виходить ще якесь ребро $r > b$. Тоді зірка з центром в v буде кращою відповіддю, ніж наш трикутник:

$$\text{зірка} \geq l_2 + l_3 + r > l_2 + l_3 + l_1 = \text{трикутник}.$$

2. Розбиття на три

Блок 1 ($n = 3$)

Очевидно, існує єдине можливе розбиття на три підмасиви, кожен розміром один елемент.

Блок 2 ($a_i \leq 1$)

Спочатку вирішимо, які будуть суми в підмасивах. Нехай змінна cnt означає кількість одиничок у масиві. Логічно кожен підвідрізок зробити за сумою, якомога ближчою до $\left\lfloor \frac{cnt}{3} \right\rfloor$. Якщо $cnt \mid 3$, то все добре і ми знайшли суми. Інакше у нас може бути ще 1 чи 2 залишкові одиничи, які ми не розподілили нікуди. Якщо вона одна, додаємо її в будь-який підмасив, а якщо їх дві — розподіляємо їх у будь-які два різні підмасиви.

Коли ми знаємо суми, просто проходимось по масиву і додаємо наступний елемент у поточний підмасив. Якщо досягли його суми (а пропустити її ми не можемо, бо $a_i \leq 1$), зачиняємо його і відкриваємо наступний. Таким чином знаходимо межі підмасивів:

$$a + b = c.$$

Блок 3 (гарантується, відповідь 0)

Порахуємо повну суму масиву, нехай це sum. Додаткові обмеження цієї групи означають, що є таке розбиття, де всі три підмасиви мають суму $\frac{sum}{3}$. Будь-які підмасиви з іншою сумою нас взагалі не цікавлять.

Для кожного індексу i порахуємо такий мінімальний індекс $next_i \leq n$ (якщо він існує), що сума на відрізку $[a, b)$ дорівнює $\frac{sum}{3}$. Якщо не існує, запам'ятаємо це.

Проходимось по масиву, перебираючи початок першого підмасиву — i. Якщо не існує $next_i$, то цей початок точно не підходить, пропускаємо. Тоді нехай $j = next_i$. Аналогічно, обов'язково треба, аби $next_j$ існував, бо інакше таке розбиття не підходить. Якщо ж він існує, ми знайшли всі три межі між підвідрізками. Перші два з них мають суму $\frac{sum}{3}$, отже, і третій має таку саму, тобто ми знайшли відповідь. Складність — $\mathcal{O}(n)$.

Блок 4 ($n \leq 100$)

Повністю переберемо три початки підмасивів, як це запропоновано у форматі вихідних даних. Коли маємо межі x, y, z, рахуємо суми просто проходом по масиву чи знаходимо префіксні суми. Складність — $\mathcal{O}(n^4)$ чи $\mathcal{O}(n^3)$, залежно від того, який метод обрати.

Надалі всі суми будемо рахувати за допомогою префіксних сум.

Блоки 5 ($n \leq 2000$) та 6 ($n \leq 5000$)

Переберемо лише перші два індекси. Наразі з їхньою допомогою ми розбили масив на два підвідрізки. Один із них залишаємо, і його суму рахуємо, нехай це буде s. Тепер нам залишилось розбити другий підвідрізок ще на два: префікс і суфікс. Зазначимо, що багато з цих розбиттів точно не оптимальні. Інтуїтивно, нам цікаво розбити масив на два, максимально рівні за сумою. Теоретично, нехай ми розбили таким чином, що суми префікса і суфікса дорівнюють $pref$ і $suff$ відповідно. Без втрати загальності, нехай: $pref \leq suff$. Припустимо, що один елемент масиву (той, що зараз на межі) можна перевести з суфікса в префікс (те саме, що змістити межу на один), і при цьому нерівність залишиться правильною. Формально: $pref + x \leq suff - x$. Нам завжди буде вигідно зробити цей перехід, тому що $\min(s, pref) \leq min(s, pref + x)$ і $\max(s, suff) \geq max(s, suff - x)$, а отже, відповідь буде не гіршою.

Єдиний момент, коли нам може бути не вигідно робити перехід, це коли знак в нерівності змінюється. Тож треба фактично перевірити лише два випадки: найбільший префікс, сума якого менша за суму суфікса, та, навпаки, найбільший суфікс, сума якого менша за суму префікса. Зі всіх інших комбінацій префікса й суфікса можна перейти в ці два випадки, не погіршуючи відповідь.

Перший варіант реалізації: після перебору перших двох індексів бінарним пошуком шукаємо переломний момент в сумах і перевіряємо ці два випадки. Складність — $\mathcal{O}(n^2 \times \log n)$, проходить п'яту групу.

Другий варіант: переломний момент шукаємо другим вказівником, який завжди рухається праворуч циклічним масивом. Складність — $\mathcal{O}(n^2)$, проходить шосту групу.

Блоки 7 ($n \leq 10^5$) та 8 (без додаткових обмежень)

Нехай сума всіх елементів масиву це sum. Зауважимо, що в будь-якому розбитті рівно одне з наступних двох тверджень правдиве:

- рівно два масиви мають суму $\leq \left\lfloor \frac{sum}{3} \right\rfloor$ і третій має суму $\geq \left\lfloor \frac{sum}{3} \right\rfloor$;
- рівно два масиви мають суму $\geq \left\lfloor \frac{sum}{3} \right\rfloor$ і третій має суму $\leq \left\lfloor \frac{sum}{3} \right\rfloor$.

Переберемо розріз масиву між двома підмасивами, що обидва мають суму, не більшу за третину sum. Аналогічно до доведення з попередньої групи, з усіх таких масивів вигідно взяти найбільші за сумою з обох сторін. Це можна зробити за допомогою бінарного пошуку або двома вказівниками. Суму третього відрізка рахуємо як залишок.

Так само перебираємо розріз між двома підмасивами, що обидва мають суму, більшу за третину sum, проте тепер мінімізуємо ці два підмасиви. Кінцева складність — $\mathcal{O}(n \times \log n)$ з бінпошуком (7 група) та $\mathcal{O}(n)$ зі вказівниками (повне рішення).

З точки зору реалізації допомогти подвоїти масив може вставлення в кінець копії початкового. Тоді замість того, щоб переходити з останнього елемента до початкового, можна просто продовжувати обхід масиву, єдине що треба буде перевірити, це те, чи не накладається кінець третього підмасиву на перший.

3. Випуклий масив

Твердження. У випуклому масиві різниці між сусідніми елементами не зменшуються. Тобто, якщо позначити $d_i = b_{i+1} - b_i$, то для кожного i виконується нерівність $d_i \le d_{i+1}$.

Це безпосередньо випливає з визначення випуклості.

Оскільки послідовність різниць $d_i = b_{i+1} - b_i$ є неспадною, існує такий індекс k, що $d_i \le 0$ для $i < k$ і $d_i \ge 0$ для $i \ge k$. Це означає, що масив можна розбити на дві монотонні випуклі послідовності: спадну $b_1, \dots, b_k$ і зростаючу $b_k, \dots, b_n$.

Тоді потрібно розбити масив на дві зростаючі випуклі послідовності, які перетинаються у найменшому елементі масиву.

Логічно будувати ці послідовності, додаючи елементи у порядку зростання. Але як визначити, до якої з них приєднати новий елемент?

Для цього достатньо зберігати по два останні елементи кожної з послідовностей. Це дасть змогу перевіряти умови випуклості та правильно розподіляти нові елементи.

Тоді можна написати динаміку $dp[i][a][b][c]$ ($i > a$, $i > b > c$), яка буде приймати значення *true* або *false* і позначати, чи можемо розділити перші i чисел на дві послідовності. Перша послідовність закінчується на елементах з індексами i, a, а друга — на b, c. Тоді маємо рішення за $O(n^4)$.

Звернемо увагу, що в досяжних станах рівно одне з a, b, c буде рівне $i - 1$. Тому насправді можна оптимізувати попередню динаміку до $O(n^3)$.

Крім того, замість бінарних значень *true* та *false* для трійки i, a, b можна зберігати найбільше можливе значення c, оскільки вигідно мінімізувати різницю між b та c. І, таким чином, матимемо рішення за $O(n^2)$.

Для того, щоб оптимізувати розв'язок, потрібно ще глибше зануритися в особливості станів. Зокрема, ми вже розглянули, що індекс $i - 1$ є важливим у станах. Тепер розглянемо також індекс $i - 2$.

Є три випадки станів $i - a, b, c$:

1) $i, i-1, b, c$;
2) $i, i-2, i-1, c$;
3) $i, c, i-1, i-2$.

Для другого і третього випадків достатньо зберегти лише максимальні можливі досяжні значення c для фіксованого i. Те ж саме

можемо зазначити і стосовно першого випадку для фіксованих i та b, але тут більше параметрів, тому цей випадок є складнішим.

Розглянемо переходи для фіксованого i. Маємо не більше ніж один стан другого та третього типів. Станів першого типу може бути багато, тому аналізуватимемо їх окремо, щоб оптимізувати переходи з цих станів.

Зокрема, стан $i, i-1, b, c$. Зосередимося на двох наведених нижче випадках залежно від того, в яку послідовність ставимо елемент $i+1$.

1. Додаємо в першу підпослідовність. Тоді має виконуватись умова: $a_{i+1} + a_{i-1} \geq 2 \times a_i$. І переходимо в стан: $i+1, i, c, d$. Зазначимо, що в цьому стані перехід відбувається незалежно від значень c, d, і перехід відбувається зі стану першого типу до стану першого типу.

Тобто якщо для фіксованого i зберігати всі можливі стани як множину пар (c, d), то при переході до наступного стану $i+1$ всі ці стани залишаться в такому ж вигляді, якщо умова $a_{i+1} + a_{i-1} \geq 2 \times a_i$ виконується.

2. Додаємо елемент $i+1$ в другу підпослідовність. Тоді має виконуватись умова: $d + a_{i+1} \geq 2 \times c$, що еквівалентно $a_{i+1} \geq 2 \times \times c - d$. І переходимо в стан: $i+1, c, i, i-1$, тобто у стан третього типу.

Для станів третього типу потрібно максимізувати c, тому необхідно знайти максимальне c серед усіх станів, що задовольняють умову $a_{i+1} \geq 2 \times c - d$.

Переходи зі станів другого і третього типів оптимізувати не треба, бо їх не більше одного.

Тепер залишилося оптимізувати переходи зі стану першого типу до третього. Для цього можна відсортувати стани (c, d) за значенням $2 \times c - - d$. Після цього для кожного нового елемента a_{i+1} потрібно знаходити максимальне значення c серед тих станів, де $2 \times c - d \leq a_{i+1}$.

Щоб ефективно знаходити цей максимум, немає необхідності використовувати складні структури даних, такі як дерево відрізків або дерево Фенвіка. Достатньо *std::set*, якщо розуміти, що стани, для яких існує інший стан з меншим $2 \times c - d$ і водночас більшим c, не потрібно зберігати.

Отже, якщо відсіяти невигідні стани, то отримані стани будуть відсортовані як за $2 \times c - d$, так і за c. Тоді для знаходження максимуму на префіксі достатньо одного виклику *lower_bound*.

Фінальна складність рішення: $\mathcal{O}(n \log n)$.

II тур

4. Проста підпослідовність

Блок 1

Ans = R – L + 1 – (L%2==0) – (R%2==0) + ((L==R)&&(L%2==0))

Блок 2

Можна перебрати всі 2^n підпослідовностей і перевірити, чи вони є хорошими, порахувавши всі префіксні і суфіксні суми. Можна для кожного з $\frac{n(n+1)}{2}$ можливих запитів порахувати відповідь, перебравши всі її підпослідовності. Складність $\mathcal{O}(2^n \times n)$.

Блок 3

Нехай $\sum[L,\ R]$ означає суму на підвідрізку $[L,\ R]$. Скажемо, що $\sum[L, R] \overset{def}{=} 0$ за $L > R$ (порожній відрізок).

Нехай $\sum[1,\ n] = S$.

Визначення хорошого масиву переписується як: $\sum[1,\ a] \geq 0$, $\sum[a,\ n] \geq 0$. Але $\sum[a,\ n] = \sum[1,\ n] - \sum[1, a-1]$. Тому визначення хорошого масиву можна записати як: $\sum[1,\ a] \in [0,\ S]$.

Тож можна обрахувати *dp[позиція][поточна сума][максимальна префіксна сума, яка зустрічалася]*, де значення динаміки — це максимальна кількість елементів (або *–INF*, якщо стан недосяжний). Складність $\mathcal{O}(qn^3)$.

Також зазначимо, що оптимальна відповідь отримується шляхом видалення з заданого відрізка певної кількості мінус одиничок. Наша мета — з'ясувати, скільки мінус одиничок ми маємо видалити.

Блок *

Твердження. Масив є хорошим тоді і тільки тоді, коли всі суми його підвідрізків є не більшими за суму всього масиву.

Доведення.

⇐: Від супротивного, припустимо, всі відрізки мають суму, меншу за S, але масив не є хорошим. Через те, що масив не є хорошим, то існує або його префікс, або суфікс, із від'ємною сумою. Без обмеження загальності, нехай це префікс, тобто існує R таке, що: $\sum[1,\ R] < 0$. Але тоді $\sum[R+1,\ n] = S - \sum[1,\ R] > S$, тобто відрізок $[1,\ R]$ має суму, більшу за суму всього масиву.

⇒: Від супротивного, припустимо, що масив є хорошим, але існує $\sum[L,\ R] > \sum[1,\ n]$. З огляду на це $\sum[1,\ n] - \sum[L,\ R] = \sum[1,\ L-1] + \sum[R+1,\ n] < 0$, тому хоча б одне з чисел $\sum[1,\ L-1]$ або $\sum[R+1,\ n]$ є від'ємним, бо їх сума від'ємна. Отже, масив не є хорошим.

Повернемося до задачі — потрібно визначити мінімальну кількість «–1», яку необхідно видалити, щоб суми на всіх підвідрізках не перевищували суму на всьому масиві.

Нехай $\sum[1,\ n] = S_0$. Нехай $[L,\ R]$ — це підвідрізок з максимальною сумою (якщо таких декілька, то довільний з них). Нехай $\sum[L,\ R] = M_0$. Хочемо зробити, щоб $S = M$, але спочатку маємо $M_0 \geq S_0$.

За одне видалення мінус одинички S збільшується на 1, а M не зменшується. Отже, різниця $M - S$ зменшується не більше ніж на 1. Тож ми маємо видалити щонайменше $M_0 - S_0$ елементів.

Доведемо, що $M_0 - S_0$ елементів видалити завжди достатньо. Нехай $\sum[1,\ L-1] = -P$ і $\sum[R+1,\ n] = -Q$. Зазначимо, що $M_0 - S_0 = \sum[L,\ R] - \sum[1,\ n] = -(\sum[1,\ L-1] + \sum[R+1,\ n]) = P + Q$.
Зауважимо також, що $P,\ Q \leq 0$ (тобто $\sum[1,\ L-1] \leq 0$ та $\sum[R+1,\ n] \leq 0$), бо інакше $[L,\ R]$ можна розширити зі збільшенням суми.

Нехай $\sum'[L,\ R]$ — це сума на підвідрізку після нашого видалення. Ми хочемо зробити, щоб тепер всі суми $\sum'[1,\ a]$ та $\sum'[a,\ n]$ були невід'ємними. Для цього видалимо $P + Q$ мінус одиничок.

До того ж видалимо на префіксі $[1,\ L-1]$ деякі P мінус одиничок, а на суфіксі $[R+1, n]$ деякі Q мінус одиничок. Які саме — з'ясуємо пізніше. Тоді $\sum'[1,\ L-1] = \sum[1,\ L-1] + P = -P + P = 0$, аналогічно $\sum'[R+1,\ n] = 0$.

Зазначимо, що $\sum[a+1,\ L-1] \leq 0$, бо інакше $\sum[a+1,\ R] = \sum[L,\ R] + \sum[a+1,\ L-1] > \sum[L,\ R]$, а ми позначили $[L,\ R]$ як відрізок з найбільшою сумою.

Візьмемо до уваги, що $\sum[L,\ a] \geq 0$, бо інакше $\sum[a+1,\ R] = \sum[L,\ R] - \sum[L,\ a] > \sum[L,\ R]$, а ми позначили $[L,\ R]$ як відрізок з найбільшою сумою.

Визначимо, які додаткові умови необхідні для того, щоб отриманий масив був хорошим:

1. $a < L$:
   З'являється умова $\sum'[1,\ a] \geq 0$.

2. $L \le a \le R$:
$\sum'[1,\ a] = \sum'[1,\ L] + \sum'[L,\ a] = 0 + \sum'[L,\ a] \ge 0$,
тобто це виконується автоматично.
3. $a > R$:
$$\sum{}'[1,a] = \sum{}'[1,L-1] + \sum{}'[L,R] + \sum{}'[R+1,a] =$$
$$= 0 + M_0 + \sum'[R+1,a] = M_0 + \sum'[R+1,n] - \sum'[a+1,n] =$$
$$= M_0 + 0 - \sum'[a+1,\ n],$$
тобто з'являється умова $\sum'[a+1,\ n] \le M_0$.

Отже, необхідно і достатньо, щоб для всіх $a < L$ виконувалось $\sum[1,\ a] \in [0,\ M_0]$, а також аналогічна умова для суфіксів, тобто $\sum[a,\ n] \in [0,\ M_0]$ для всіх $a > R$. Цього, наприклад, можна домогтися, якщо видаляти «–1»: йдемо по $[1 \ldots a]$, збільшуючи a від 0 до $L-1$, і, якщо префіксна сума стала від'ємна, то видаляємо поточну мінус одиницю. Тоді всі префіксні суми стануть невід'ємні. Припустимо, що при видаленні одиниці на позиції a виявилось, що $\sum'[1,\ b]$ стала більшою за $M_0 + 1$ для певного $b < L$. Зауважимо, що в цей момент змінились тільки префіксні суми, які стоять правіше за a, тобто $b \ge a$. Але $\sum'[1,\ a]$ у цей момент стала дорівнювати 0. Тому $\sum'[a+1,\ b] = \sum'[1,\ b] - \sum'[1,\ a] = M_0 + 1$. Але на цьому відрізку ми нічого не змінювали, тобто $\sum'[a+1,\ b] = \sum[a,\ b] = M_0 + 1 > M_0 = \sum[L,\ R]$.

Однак ми обирали M_0 як максимальну суму на підвідрізку. Тому такого не може бути, а значить, усі префіксні суми знаходяться у потрібних межах. Аналогічно робимо з суфіксами.

Варто зазначити, що ми дійсно видалили рівно P мінус одиничок, тобто що сума на всьому префіксі стала дорівнювати нулеві. Дійсно, якщо після видалення мінус одиничи з позиції a виявилось, що $\sum'[1,\ L-1] > 0$, то $\sum[a+1,\ L-1] = \sum'[a+1,\ L-1] = \sum'[1,\ L-1] - \sum'[1,\ a] = \sum'[1,\ L-1] - 0 > 0$, а ми доводили, що суми на відрізках виду $[a, L-1]$ завжди невід'ємні.

Отже, ми звели задачу до пошуку M_0, S_0 і виведення на екран величини $R - L + 1 - (M_0 - S_0)$.

Блок 4

В розборі цього і наступних блоків описано, як знайти:

$M = \max\limits_{l \le L \le R \le r} \sum[L,\ R]$.

Як показано вище, це дасть змогу розв'язати задачу.

Переберемо R. Маємо обрати L таке, що $\sum[L,\ R]$ максимальне, тобто $\sum[1,\ L-1]$ мінімальне. Можемо підтримувати це L одночасно з перебором R.

Складність: $\mathcal{O}(qn)$

Блоки 5–6

Нехай $node\ [L,\ R]$ — це четвірка значень:

- S — сума на цьому відрізка;
- A — максимальна сума префікса цього відрізка;
- B — максимальна сума суфікса цього відрізка;
- M — максимальна сума певного підвідрізка цього відрізка.

Тоді можна рахувати $node\ [L,\ R]$ через $node\ [L,\ a]$ та $node\ [a+1,\ R]$ для певного a. Нехай:

$(S_1, A_1, B_1, M_1) + (S_2, A_2, B_2, M_2) = (S, A, M, B)$,

тоді:

- $S = S_1 + S_2$;
- $A = \max(A_1, S_1 + A_2)$;
- $B = \max(B_2, S_2 + B_1)$;
- $M = \max(M_1, M_2, B_1 + A_2)$.

Якщо ви дуже хочете здати ці блоки, але не здати наступний, можна використати sqrt-декомпозицію за $\mathcal{O}(q\sqrt{n})$. Для цього слід розділити масив на блоки довжини C (де $C \sim \sqrt{n}$) і підрахувати $node\ [kC,\ (k+1)C-1]$. Далі для кожного запиту пошуку відповіді можна виконувати $\mathcal{O}(n/C)$ описаних вище злиттів, а для запитів оновлення потрібно перераховувати суму для зміненого блоку за $\mathcal{O}(C)$.

Блок 7

Описане в попередньому блоці злиття можна обраховувати в дереві відрізків, у вершинах якого будуть зберігатись числа S, A, B, M для відповідних відрізків. Запит *get* також буде використовувати це злиття і поверне четвірку чисел, з якої ми візьмемо M та S і виведемо $R - L + 1 - -(M - S)$.

6. Подарунок для Антона

Блок 0

Якщо є хоча б одна «4», то всі поруч із нею теж «4», поруч із цією «4» — теж всі «4» і так далі. А таблиця скінченна, отже, такого не може бути.

Якщо є хоча б одна «3», то розглянемо найвищу з усіх «3». Тоді у всіх напрямках, окрім верхнього, вона межує з іншими «3». Розглянемо ту «3», що справа від початкової. В неї аналогічно зліва, знизу та справа інші «3». І так далі, можемо розширюватись вправо до нескінченності, а таблиця скінченна — суперечність.

Двійки утворюють циклічні структури. Одинички розташовані доміношками. Нулі є ізольованими від інших нулів. Це яскраво видно з умови задачі.

Описані вище роздуми не є обов'язковими, але подібний аналіз завдання дуже допомагає у розв'язанні.

Блок 1 ($n = 1$)

Нескладно помітити, що за допомогою чергування доміношок «1-1» та ізольованих «0» можна отримати ланцюг довільної довжини.

Блок 2 ($n = 2$)

Якщо продублювати рішення на блок 1 і збільшити всі числа на 1, то воно буде працювати на цей блок.

Блок 2 ($n = 3$)

Якщо обрамлення зробити з «2», то лишиться стрічка $1 \times (m - 2)$, для якої можна викликати рішення для блоку 1.

Блок 3

Візьмемо приклад з умови задачі. Відріжемо найлівіший і найправіший стовпці. Кути поміняємо з «1» на «0».

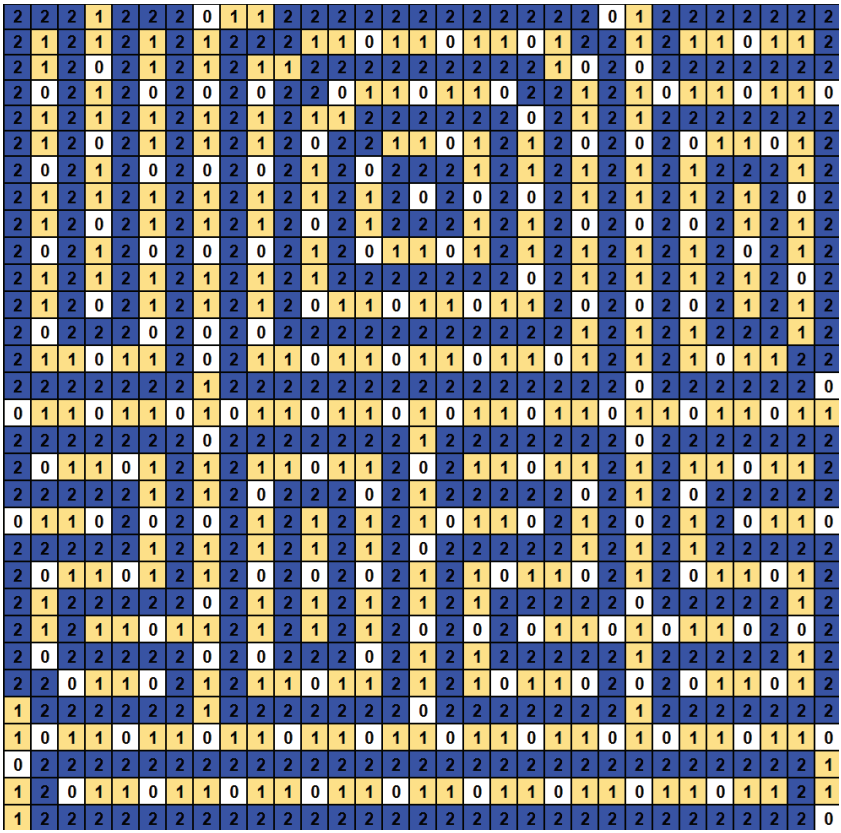
Блоки 5–11

Якщо поставити поруч із квадратиком з «2» дві доміношки з «1» та ізольований «0», то отримаємо паттерн 3×3, який можна повторювати нескінченно.

2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
2	2	1	2	2	1	2	2	1	2	2	1	2	2	1	2	2	1
1	1	0	1	1	0	1	1	0	1	1	0	1	1	0	1	1	0

Ба більше, якщо розглянути конкретний рядок, то помітимо, що в нас завжди наявна така ситуація, як у блоці 1: дві однакові цифри, одна інша, дві однакові, одна інша... Це значить, що так само, як і в блоці 1, ми можемо обрати відрізок з m стовпців, щоб на границях не було проблем. Аналогічно можна обрати відрізок з n рядків, щоб на границях не було проблем. Відповідь — таблиця, що утворена перетином цих n рядків та m стовпців.

Блок 8



7. Розворот ABC

Блок 1

У цьому блоці відповідь може бути 0 або 1. Щоб його розв'язати, потрібно перевірити, чи можливо виконати хоча б одну операцію. Якщо так — вивести 1, інакше — 0.

Необхідно з'ясувати, чи існує таке i, що підрядок $s_i s_{i+1}$ дорівнює одному з трьох: AB, BC або CA.

Блок 2

У цьому блоці потрібно перебрати всі можливі рядки довжини до 3 і побудувати дерево розгалужень.

Блок 3

У цьому блоці рядок складається лише з символів A та B, отже, можлива тільки одна операція — над підрядком AB.

Фінальний вигляд рядка — це всі B ліворуч, а всі A праворуч: $BB...BBAA...AA$, бо більше неможливо виконати жодної операції.

Щоб порахувати кількість операцій, для кожного символу A потрібно підрахувати кількість B, що йдуть після нього, та просумувати для всіх A.

Блок 4

Для рядка виду $AA...AABB...BBCC...CC$, нехай n — кількість A, m — кількість B, k — кількість C.

Вигідно або замінити всі A та B, або всі B та C.

Відповідь: $\max(n \times m, m \times k)$.

Блок 5

Погрупуємо однакові підрядки символів у початковому рядку — вони виконуватимуть операції разом.

Якщо підрядок CA ніколи не з'являвся, він і не з'явиться після жодної кількості операцій.

Доведення. Якщо припустити, що CA з'явився, то до цієї операції рядок мав містити CBA або AC, але з жодного з цих рядків не утворюється CA.

Можливі лише операції над AB та BC. Для кожного символу B розглянемо такі випадки:

1) B має зліва лише A: взаємодіє з усіма A до першого C зліва;

2) B має справа лише C: взаємодіє з усіма C до першого A справа;

3) B має A зліва і C справа: обираємо максимум з випадків 1 і 2.

Блок 6

Якщо довжина рядка ≤ 12, можна запускати повну рекурсію.

Якщо операції виконати неможливо — відповідь 0.

Інакше перебираємо всі можливі операції, запускаємо рекурсію для кожного варіанта і вибираємо максимальну відповідь.

Складність: $O(n \times 3^n)$, оскільки кількість станів — 3^n, і з кожного стану — до n переходів.

Блок 7

Щоб розв'язати цей блок, попередньо порахуємо відповідь для всіх 3^n станів (як у блоці 6).

Після цього на кожен запит відповідатимемо за $O(1)$.

Блок 9

Погрупуємо однакові підрядки символів у початковому рядку — вони діятимуть разом.

Результуючий рядок матиме вигляд *...ACBACBACB...*.

Введемо dp[перший символ][суфікс] — відповідь для суфікса, якщо перший символ у результаті — вказаний.

Тоді шукаємо $\max(dp['A'][0], dp['B'][0], dp['C'][0])$.

Переходи в dp, для прикладу $first_{symbol}$ = A (аналогічно для інших):

- якщо $s[i] = \text{A}$, то $dp[\text{A}][i] = \max(dp[\text{A}][i+1], dp[\text{C}][i+1])$ (після *A* не може бути *B*);
- якщо $s[i] = \text{B}$, то $dp[\text{A}][i] = -\infty$;
- якщо $s[i] = \text{C}$;
- якщо далі йде *B* або кінець рядка, то $dp[\text{A}][i] = -\infty$;
- якщо рядок виглядає як *CACACACA[C]*, то переміщуємо всі *A* ліворуч:
  $dp[\text{A}][i] = \text{value\_of\_swapping\_all\_A\_to\_left}$;
- якщо рядок виглядає як *CACACACACB...*:
  $dp[\text{A}][i] = \text{value\_of\_swapping\_all\_A\_to\_left} + dp[\text{B}][B\dots]$;
- якщо рядок має вигляд *CACACACAB...*, то:

$dp[\text{A}][i] = \max(\text{value\_of\_swapping\_all\_A\_to\_left} + dp[\text{B}][B\dots],$ $\text{value\_of\_swapping\_bold\_A\_to\_left} + dp[\text{B}][AB\dots])$.

8. Проста задача

Введемо наступні позначення:

- $x = \max\limits_{i=1}^{n} a_i$;
- $N = \sum_{i=1}^{T} n_i$, де n_i — значення n в i-му наборі вхідних даних;
- $X = \max\limits_{i=1}^{T} x_i$, де x_i — значення x в i-му наборі вхідних даних;
- ans — мінімально можлива кількість *дивних* підпослідовностей серед k утворених.

Блок 1 ($k = 1$)

Існує єдиний спосіб розбити масив a на одну підпослідовність. Щоб знайти кількість *дивних* підпослідовностей серед однієї утвореної, достатньо перевірити, чи є сума елементів масиву a простим числом — це можна зробити стандартним алгоритмом за $O(\sqrt{s})$, де $s = \sum_{i=1}^{n} a_i \le nx$.

Складність оброблення одного набору вхідних даних — $O(n + \sqrt{nx})$.

Складність оброблення всіх наборів вхідних даних — $O(N + N\sqrt{X})$.

Блоки 2, 3, 4, 5, 6, 7, 8

Для розв'язання наступних блоків доведемо дві допоміжні леми.

Лема 1

Твердження. Нехай $q = n - k$. Розглянемо наступний процес:

• створимо n підпослідовностей, де i-та початково складається з одного елемента a_i;

• виконаємо q операцій «об'єднання»: під час кожної операції виберемо дві певні підпослідовності та об'єднаємо їх в одну.

Тоді задача розбиття масиву на k непорожніх підпослідовностей еквівалентна задачі об'єднання n одноелементних підпослідовностей за допомогою q описаних вище операцій.

Доведення. Дійсно, при виконанні однієї операції кількість підпослідовностей зменшується на 1, а отже, після виконання q операцій кількість підпослідовностей буде дорівнювати $n - q = n - n + k = k$. Також легко помітити, що будь-яке розбиття можна утворити за допомогою q описаних вище операцій: для цього під час виконання чергової операції потрібно вибрати дві довільні підпослідовності, елементи яких належать одній у відповідному розбитті.

Лема 2

Твердження. $ans \geq n - 2q$.

Доведення. Дійсно, в описаному вище процесі початково існувало n *дивних* підпослідовностей, а кожна операція «об'єднання» зменшувала кількість *дивних* підпослідовностей не більше ніж на 2. Тому після виконання будь-яких q операцій все ще мало б залишитися хоча б $(n - 2q)$ *дивних* підпослідовностей.

Блок 6 $(2k \geq n + 1)$

Оскільки $k = n - q$, то $2 \times (n - q) \geq n + 1 \Rightarrow n - 2q \geq 1$. У такому випадку завжди можна утворити відповідь рівно з $(n - 2q)$ *дивними* підпослідовностями, що є оптимальною відповіддю згідно з другою лемою. А саме, під час виконання чергової операції «об'єднання» виберемо дві одноелементні підпослідовності з однаковою парністю елементів. Неважко помітити, що це завжди можна буде зробити:

• для непарного n існує парна кількість або парних, або непарних елементів, а тому максимум один елемент може залишитися без пари, що відповідає умові $(n - 2q) \geq 1$;

• для парного n максимум два елементи можуть залишитися без пари (один парний, інший непарний), що відповідає умові $(n - 2q) \geq 2$, яка виконується, оскільки $(n - 2q)$ є парним числом та $(n - 2q) \geq 1$.

Оскільки сума двох чисел з однаковою парністю є парним числом, більшим за два, то у результаті «об'єднання» замість двох *дивних* підпослідовностей утвориться одна *недивна*. Отож кожна операція буде зменшувати кількість *дивних* підпослідовностей на 2, що в підсумку зробить $ans = n - 2q$.

Складність оброблення всіх наборів вхідних даних — $O(N)$.

Блоки 2, 3, 7

Перед обробленням всіх наборів вхідних даних підрахуємо допоміжний масив *isPrime[1..2X]* (*isPrime[1..10X]* у третьому блоці), де *isPrime[i]* позначає, чи є число i простим. Це можна зробити за $O(X\log X)$ за допомогою алгоритму «Решето Ератосфена».

Блок 2 ($n \leq 4$)

За наявності рішення на перший та шостий блоки потрібно розглянути тільки випадок $n = 4, k = 2$, оскільки для всіх інших випадків виконується хоча б одна з умов $2k \geq n + 1$, $k = 1$.

За $n = 4, k = 2$ достатньо розглянути лише випадки, де обидві підпослідовності складаються з двох елементів. Це можна зробити, перебравши один з трьох елементів, який буде в підпослідовності разом з $a[1]$. Якщо в якомусь із трьох випадків немає *дивних* підпослідовностей (це можна перевірити за допомогою масиву *isPrime*), то $ans = 0$. Інакше серед цих випадків треба знайти будь-який з однією *дивною* підпослідовністю. Такий завжди знайдеться, оскільки в масиві довжини 4 існує пара елементів однакової парності (а отже, їхня сума не є простим числом). Саме тому не потрібно розглядати випадки, де одна підпослідовність складається з одного елемента, а інша — з трьох, оскільки в них гарантовано буде хоча б одна *дивна* підпослідовність (яка складається з одного елемента, що є простим числом).

Не маючи рішень на перший та шостий блоки, можна окремо розглянути усі можливі комбінації значень n, k, та розв'язати задачу для кожної з них окремо, перебравши усі необхідні варіанти утворення підпослідовностей.

Складність оброблення всіх наборів вхідних даних — $O(X\log X + N)$.

Блок 3 ($T \leq 20, n \leq 10$)

У цьому блоці достатньо було перебрати всі можливі розбиття масиву a на k підпослідовностей. А саме, будь-яке розбиття можна задати за допомогою масиву ids довжини n, де $ids[i]$ позначає номер підпослідовності, до якої належить $a[i]$.

Щоб розбиття однозначно задавалося масивом ids, достатньо розглядати лише такі ids, де $ids[1] = 1$ та $1 \leq ids[i] \leq \left(\max_{j=1}^{i-1} ids[j] + 1\right)$ при $i > 1$. Дійсно, оскільки порядок підпослідовностей не має значення, без втрати загальності можна вважати $ids[1] = 1$, а для кожного наступного i i-й елемент або буде входити до підпослідовності, в якій вже зустрічався якийсь попередній елемент масиву a (тоді $1 \leq ids[i] \leq \max_{j=1}^{i-1} ids[j]$), або буде найлівішим елементом нової підпослідовності (тоді без втрати загальності можна вважати $ids[i] = \max_{j=1}^{i-1} ids[j] + 1$).

Усі розбиття можна перебрати за допомогою рекурсивної генерації масиву ids. Наприклад, наступний код на *Python3* перебирає та рахує кількість розбиттів масиву довжини 10:

```
def countPartitions(ids):
if len(ids) == 10:
return 1
return sum(countPartitions(ids + [id]) for id in range(1, max(ids) + 2))
print(countPartitions([1]))
```

Експериментальним шляхом можна з'ясувати, що за $n = 10$ кількість розбиттів $C = 115975$. Для кожного з них із загальною кількістю підпослідовностей k у розбитті можна підрахувати кількість *дивних* підпослідовностей за допомогою масиву *isPrime* та вибрати розбиття з мінімальною кількістю.

Складність оброблення всіх наборів вхідних даних — $O(X\log X + + C \times N)$.

Блоки 4, 5, 7, 8

Для цих блоків буде наведено рішення для $1 < k$, $5 \le n$ та $2k < n + 1 \Rightarrow 2k \le n \Rightarrow n \le 2q$. За невиконання цих умов дивіться рішення для блоків 1, 2 та 6.

Блок 4 (n — парне, $a_i > 2$)

З огляду на те. що всі $a_i > 2$, то всі елементи масиву a — непарні. Оскільки $q \ge \frac{n}{2}$, то спочатку розіб'ємо усі елементи на пари, тим самим виконавши $\frac{n}{2}$ операцій «об'єднання» та зробивши всі підпослідовності *недивними*, тому що суми їхніх елементів будуть парними числами, більшими за два. Після цього залишиться виконати $(q - \frac{n}{2})$ операцій «об'єднання», що можна зробити довільним чином, оскільки сума будь-яких двох парних чисел є парним числом. Кількість *дивних* підпослідовностей у результаті буде дорівнювати нулю, що очевидно є мінімально можливою кількістю.

Складність оброблення всіх наборів вхідних даних — $O(N)$.

Блоки 5, 7, 8

Для розв'язання наступних блоків доведемо допоміжну лему.

Лема 3

Твердження. Серед п'яти простих чисел завжди можна знайти три числа, сума яких ділиться на 3 та не є простим числом.

Доведення. Зазначимо, що сума будь-яких трьох простих чисел перевищує 3, а отже, ця сума не є простим числом у випадку, якщо вона ділиться на 3. Тепер доведемо, що можна знайти три числа, сума яких ділиться на 3. Припустимо, що це не так. Розглянемо залишки від ділення на 3 всіх заданих чисел. Якщо якийсь залишок зустрічається хоча б 3 рази, то сума трьох відповідних чисел буде ділитися на 3, що за припущенням неможливо. Також якщо усі залишки (0, 1, 2) зустрічаються хоча б один раз, то сума трьох відповідних чисел буде ділитися на 3, що за припущенням неможливо. Це означає, що у масиві залишків спостерігається лише два різних числа, кожне з яких зустрічається максимум два рази, а тому загальна довжина масиву не перевищує 4. Отримали протиріччя, оскільки лема доводилася для п'яти чисел.

Блок 5 (n — непарне, $a_i > 2$)

Оскільки q — ціле число, то для непарного n нерівність $q \geq \frac{n}{2}$ можна посилити: $q \geq \frac{n+1}{2}$. Оскільки $n \geq 5$, то згідно з третьою лемою серед перших п'яти елементів масиву a можна знайти трійку з сумою, що не ділиться на 3. Знайдемо таку трійку шляхом повного перебору всіх $C_5^3 = 10$ трійок та об'єднаємо відповідні елементи в одну підпослідовність, виконавши дві операції «об'єднання». Інші $(n-3)$ елементи розіб'ємо на пари довільним чином, виконавши $\frac{n-3}{2}$ операцій «об'єднання». Після цих $(2 + \frac{n-3}{2} = \frac{n+1}{2})$ операцій «об'єднання» усі підпослідовності будуть *недивними* і тільки перша матиме непарну суму елементів.

Оскільки $k > 1$, то, маючи лише одну *недивну* підпослідовність з непарною сумою та решту *недивних* підпослідовностей з парною, фінальні $c = (q - \frac{n+1}{2})$ операцій «об'єднання» можна витратити на об'єднання $(c+1)$ підпослідовностей з парною сумою в одну підпослідовність також з парною сумою. Кількість *дивних* підпослідовностей у результаті дорівнюватиме нулю, що очевидно є мінімально можливою кількістю.

Складність оброблення всіх наборів вхідних даних — $O(N)$.

Блок 8 (n — непарне)

Аналогічно п'ятому блоку тут буде виконуватися нерівність $q \geq \frac{n+1}{2}$.

Поступово розглянемо всі можливі «типи» масиву a:

1. Кількість двійок у масиві a — непарна.

Тоді кількість непарних елементів у масиві a — парна. Залишимо одну двійку в окремій одноелементній підпослідовності та використаємо $\frac{n-1}{2}$ операцій «об'єднання» для розбиття решти $(n-1)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Використаємо додаткову операцію «об'єднання», щоб об'єднати одноелементну підпослідовність із двійкою з будь-якою іншою підпослідовністю. Після цих $(1 + \frac{n-1}{2} = \frac{n+1}{2})$ операцій «об'єднання» усі підпослідовності будуть *недивними* та матимуть парну суму елементів.

Фінальні $(q - \frac{n+1}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку, отримавши нульову кількість *дивних* підпослідовностей у результаті.

2. Кількість двійок у масиві a — парна.

Тоді кількість непарних елементів у масиві a — непарна. Спробуємо знайти трійку непарних елементів з сумою, що ділиться на 3 (це можна зробити так само, як у блоці 5):

- Якщо така трійка знайшлася, то об'єднаємо відповідні елементи в одну підпослідовність, виконавши дві операції «об'єднання». Виконаємо $\frac{n-3}{2}$ операцій «об'єднання» для розбиття решти $(n-3)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Після цих $(2 + \frac{n-3}{2} = \frac{n+1}{2})$ операцій «об'єднання» усі підпослідовності будуть недивними і тільки перша матиме непарну суму елементів. Фінальні $(q - \frac{n+1}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку, отримавши нульову кількість дивних підпослідовностей у результаті.
- Якщо така трійка не знайшлася, то з третьої леми випливає, що кількість непарних елементів не перевищує 3, а отже, кількість двійок хоча б 2. Розглянемо два випадки:

i. Серед непарних елементів існує якийсь елемент $x \neq 3$. Залишимо x та дві двійки в окремих одноелементних підпослідовностях. Виконаємо $\frac{n-3}{2}$ операцій «об'єднання» для розбиття решти $(n-3)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Оскільки x — просте число, більше за три, x може мати лише два можливих залишки при діленні на 3:

1. $x \bmod 3 = 2$, тобто $(x+4)$ ділиться на 3. Виконаємо дві операції «об'єднання» одноелементних підпослідовностей в одну: $[x, 2, 2]$.

2. $x \bmod 3 = 1$, тобто $(x+2)$ ділиться на 3. Виконаємо одну операцію «об'єднання», щоб створити двоелементну підпослідовність: $[x, 2]$, а другу операцію, щоб об'єднати останню двійку з будь-якою іншою з перших $\frac{n-3}{2}$ пар.

Після цих $(2 + \frac{n-3}{2} = \frac{n+1}{2})$ операцій «об'єднання» усі підпослідовності будуть *недивними*, і тільки одна матиме непарну суму елементів. Фінальні $(q - \frac{n+1}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку шляхом отримання нульової кількості *дивних* підпослідовностей у результаті.

ii. Усі непарні елементи дорівнюють трьом. Це означає, що в масиві a існує лише один непарний елемент, оскільки інакше на попередньому кроці знайшлася б трійка з сумою, що ділиться на 3 $((3 + 3 + 3) \bmod 3 = 0)$. Іншими словами, $a = [2, 2, \dots, 2, 2, 3]$. Розглянемо два випадки:

1. $q = \frac{n+1}{2}$. У такому випадку хоча б одна з утворених k підпослідовностей буде *дивною*: або підпослідовність з трійкою ($[3]$, $[3, 2]$, $[3, 2, 2]$), або одноелементна підпослідовність $[2]$ (така завжди знайдеться, якщо підпослідовність з трійкою складається хоча б із чотирьох елементів). Виконаємо $\frac{n-1}{2}$ операцій «об'єднання» для розбиття всіх $(n - 1)$ двійок на пари та ще одну операцію, щоб об'єднати $[2, 2]$ та $[3]$. Отримаємо розбиття з однією дивною підпослідовністю.

2. $q > \frac{n+1}{2} \Rightarrow q \geq \frac{n+3}{2}$. У такому випадку $n \geq 7$, оскільки за $n = 5$ єдине можливе значення $q = \frac{n+3}{2} + 1 = 4$, що вже розглянуто окремо у випадку $k = n - q = 5 - 4 = 1$.

Об'єднаємо останні чотири елементи в підпослідовність $[2, 2, 2, 3]$, виконавши три операції «об'єднання». Об'єднаємо три двійки в підпослідовність $[2, 2, 2]$, здійснивши дві операції «об'єднання». Виконаємо $\frac{n-7}{2}$ операцій «об'єднання» для розбиття решти $(n - 7)$ двійок на пари. Після цих $(5 + \frac{n-7}{2} = \frac{n+3}{2})$ операцій «об'єднання» усі підпослідовності будуть *недивними,* і тільки одна з них матиме непарну суму елементів. Фінальні $(q - \frac{n+3}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку, отримавши нульову кількість *дивних* підпослідовностей у результаті.

Складність оброблення всіх наборів вхідних даних — $O(N)$.

Блок 7 (n — парне)

Нагадаємо, що тут розглядається розв'язок для $1 < k,\ 5 \le n,\ q \ge \frac{n}{2}$. Оскільки n — парне, нерівність $n \ge 5$ можна посилити: $n \ge 6$.

Поступово розглянемо всі можливі «типи» масиву a:

1. Кількість двійок у масиві a — парна.

Тоді кількість непарних елементів у масиві a — парна. Використаємо $\frac{n}{2}$ операцій «об'єднання» для розбиття усіх n елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Після цих $\frac{n}{2}$ операцій «об'єднання» усі підпослідовності будуть *недивними* та матимуть парну суму елементів. Фінальні $(q - \frac{n}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку, отримавши нульову кількість *дивних* підпослідовностей у результаті.

2. Кількість двійок у масиві a — непарна.

Тоді кількість непарних елементів у масиві a — непарна. Спробуємо знайти непарний елемент x такий, що $(x + 2)$ не є простим числом (це можна перевірити за допомогою isPrime[x + 2]):

• Якщо такий елемент знайшовся, то об'єднаємо його з двійкою в одну підпослідовність $[x, 2]$, здійснивши одну операцію «об'єднання». Виконаємо $\frac{n-2}{2}$ операцій «об'єднання» для розбиття решти $(n - 2)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Після цих $(1 + \frac{n-2}{2} = \frac{n}{2})$ операцій «об'єднання» усі підпослідовності будуть недивними, і тільки перша матиме непарну суму елементів. Фінальні $(q - \frac{n}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку, отримавши нульову кількість дивних підпослідовностей у результаті.

• Якщо такого елемента не знайшлося, це означає, що для всіх непарних елементів x виконується $(x + 2) \bmod 3 \ne 0 \Rightarrow x \bmod 3 \ne 1$. Розглянемо два випадки:

і. $q = \frac{n}{2}$. У такому випадку хоча б одна з утворених k підпослідовностей буде *дивною*: або одноелементна підпослідовність

(яка завжди є *дивною*), або підпослідовність $[2, x]$, де $x \neq 2$ (така завжди знайдеться, оскільки двійок непарна кількість, а тому хоча б одна двійка мала б об'єднатися з непарним елементом x).

Об'єднаємо двійку з будь-яким непарним елементом та виконаємо ще $\frac{n-2}{2}$ операцій «об'єднання» для розбиття решти $(n-2)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Отримаємо розбиття з однією дивною підпослідовністю.

ii. $q > \frac{n}{2} \Rightarrow q \geq \frac{n+2}{2}$. Спочатку виконаємо перші $\frac{n+2}{2}$ операцій «об'єднання» залежно від «типу» масиву a:

1. У масиві a існує хоча б три двійки та одна трійка. Об'єднаємо ці чотири елементи в підпослідовність $[2, 2, 2, 3]$, здійснивши три операції «об'єднання». Виконаємо $\frac{n-4}{2}$ операцій «об'єднання» для розбиття решти $(n-4)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Кількість виконаних операцій «об'єднання» — $(3 + \frac{n-4}{2} = \frac{n+2}{2})$.

2. У масиві a існує хоча б п'ять двійок. У такому випадку для кожного непарного елемента x виконується $x \bmod 3 = 2$ (інакше б розглянувся попередній випадок).

Об'єднаємо дві двійки та будь-який непарний елемент x у підпослідовність $[2, 2, x]$, виконавши дві операції «об'єднання». Об'єднаємо три двійки в підпослідовність $[2, 2, 2]$, здійснивши дві операції «об'єднання». Виконаємо $\frac{n-6}{2}$ операцій «об'єднання» для розбиття решти $(n-6)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Кількість виконаних операцій «об'єднання» — $(2 + 2 + \frac{n-6}{2} = \frac{n+2}{2})$.

3. У масиві a існує рівно три двійки. Оскільки $n \geq 6$, то в масиві a існує хоча б три непарні елементи. Аналогічно попередньому випадку для кожного непарного елемента x виконується $x \bmod 3 = 2$.

Об'єднаємо три двійки в підпослідовність $[2, 2, 2]$, виконавши дві операції «об'єднання». Об'єднаємо три довільні непарні елементи x, y, z в підпослідовність $[x, y, z]$, виконавши дві операції «об'єднання» (їхня сума буде ділитися на 3, оскільки $x \bmod 3 = y \bmod 3 = z \bmod 3 = 2$).

Виконаємо $\frac{n-6}{2}$ операцій «об'єднання» для розбиття решти $(n-6)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Кількість виконаних операцій «об'єднання» — $(2+2+$ $+\frac{n-6}{2}=\frac{n+2}{2})$.

4. У масиві a існує рівно одна двійка. Оскільки $n \geq 6$, то в масиві a існує хоча б п'ять непарних елементів. Згідно з третьою лемою серед них можна знайти три елементи, сума яких ділиться на 3.

Знайдемо їх шляхом повного перебору всіх $C_5^3 = 10$ трійок та об'єднаємо відповідні елементи в одну підпослідовність за допомогою двох операцій «об'єднання». Об'єднаємо двійку з двома іншими непарними елементами x, y в підпослідовність $[2, x, y]$, здійснивши дві операції «об'єднання». Виконаємо $\frac{n-6}{2}$ операцій «об'єднання» для розбиття решти $(n-6)$ елементів масиву a на пари: парні елементи з парними, непарні — з непарними. Кількість виконаних операцій «об'єднання» — $(2+2+\frac{n-6}{2}=\frac{n+2}{2})$.

Після цих $\frac{n+2}{2}$ операцій «об'єднання» усі підпослідовності будуть *недивними*, і тільки одна з них матиме непарну суму елементів. Фінальні $(q-\frac{n+2}{2})$ операцій «об'єднання» можна зробити аналогічно фінальній частині розв'язку п'ятого блоку, отримавши нульову кількість *дивних* підпослідовностей у результаті.

Складність оброблення всіх наборів вхідних даних — $O(X\log X+N)$.

Авторська реалізація

Для кращого розуміння описаного вище розв'язку його авторську реалізацію на *Python3* можна знайти за посиланням *https://ideone.com/owiZT3*.

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Пам’ятка учасника

У розв’язку завдання дозволяється використовувати тільки файли з початковими даними, які розміщено в каталозі *«Для учасника»*.

Всі завдання необхідно виконати за 4 години. Перевірка розв’язку учасника передбачає зміну вхідних даних та перевірку результату обчислень зі зміненими даними.

Учасник має створити каталог «*Прізвище_Ім’я_Клас»* (наприклад: *Петров_Олесь_11*).

Під час роботи на олімпіаді учаснику заборонено користуватися будь-якими цифровими обчислювальними чи комунікаційними пристроями крім персонального комп’ютера на робочому місці учасника зі встановленим на ньому пакетом офісних застосунків, визначеним оргкомітетом олімпіади.

Заборонено вставляти у файли-розв’язки зображення з файлів-зразків чи з файлів-інструкцій, використовувати редактор VBA та макрорекордер за винятком вбудованих макрокоманд MS Access.

Суворо заборонено залишати в роботі відомості, які ідентифікують особу учасника (за винятком назви каталогу з файлами-розв’язками учасника).

У І турі попереднє співвідношення балів: *Access, Excel 1, Excel 2, Word, PowerPoint*: відповідно 25:10:25:15:25.

У ІІ турі попереднє співвідношення балів: *Access, Excel, Excel, Excel, PowerPoint*: відповідно 25:20:20:15:20.

І тур

Уважне вивчення умов і
допоміжних матеріалів
завдання — половина успіху

Музика — універсальна мова людства.
Генрі Уодсуорт Лонгфелло

Му́зика (від грец. μουσική — мистецтво муз) — мистецтво організації музичних звуків, насамперед у часовій (ритмічній), звуковисотній та тембровій шкалі. Музичним може бути практично будь-який звук з певними акустичними характеристиками, які відповідають естетиці тієї чи іншої епохи, що може бути відтвореним. Джерелами такого звука можуть бути: людський голос, музичні інструменти, електричні генератори тощо [1].

Звичайно, можна звузити світ спілкування з музикою до слухання улюблених музичних творів виконавців зі смартфона. І це має право на існування в сучасному мінливому, перенасиченому інформацією світі. Але справжнє враження краще отримати від живої музики, якість сприйняття якої не залежить від якості й технологій електронних пристроїв, що передають чи відтворюють її в запису. Однак не часто людина має можливість потрапити на концерт живої музики, а тому попит на записи завжди великий.

Технології запису на вінілові диски, магнітні бобіни та касети, CD та DVD за останні сто років змінювали одна одну. І сьогодні всі вони поступилися музичним ресурсам у мережі Інтернет.

Утім, зазирніть в історію музики і ви почуєте й побачите її величний світ, тих, хто створював і створює музичні шедеври, інструменти для виникнення музики, музичні події.

Автори завдання пропонують учасникам олімпіади познайомитись зі світом музики через задачі й відтворити різні інформаційні моделі об’єктів та процесів, пов’язаних із музикою.

[1] Музика. URL: https://uk.wikipedia.org/wiki/Музика (дата звернення: 25.02.2025).

Задача «MusicLabelData»

Завдання виконується виключно засобами MS Access та MS Excel. Результат роботи учасника зберегти у файл «MusicLabelData.accdb».

Музика сьогодні є не тільки одним із видів мистецтва, вона формує цілу індустрію з її власними правилами та законами. Якісь музичні гурти посідають музичний олімп роками, інші з'являються ненадовго і зникають. Як розуміти цей світ? Які виконавці і пісні популярні в певний час, які є музичні альбоми тих чи інших музичних гуртів?

Учаснику пропонується розробити аналітичну модель, у якій урахувати можливості пошуку за маскою, відображення альбомів за оцінкою музичних критиків, використати для гнучкого пошуку реальні й сценічні імена виконавців тощо. Учасник має створити інформаційну базу даних, що містить синтетично згенерований набір музичних даних для невеликої уявної музичної компанії, суворо дотримуючись інструкцій файлу *«Інструкція Access.docx»*.

Інструкція

Учаснику, відповідно до наведених нижче настанов, необхідно створити інформаційну базу даних (далі — БД), що містить синтетично згенерований набір музичних даних для невеликої уявної музичної компанії [2].

Підготовчий етап

Вихідні відомості та проєкт бази даних містяться у файлах «*MusicLabelDataset.xlsx»* та *«MusicLabelData.accdb».*

Під час виконання завдання заборонено змінювати типи полів та додавати інші поля у таблицях БД, додавати інші таблиці у БД.

Необхідно:

1. Створити ключові поля у таблицях БД.
2. Створити зв'язки з типами зв'язків між таблицями БД.
3. Імпортувати дані з файлу «*MusicLabelDataset.xlsx».*

[2] Дані взято з Kaggle. URL: https://www.kaggle.com/datasets/revilrosa/music-label-dataset/data (дата звернення: 25.02.2025).

Форма «Music Label»

Створити форму «Music Label» за зразком (*рис. 1*).

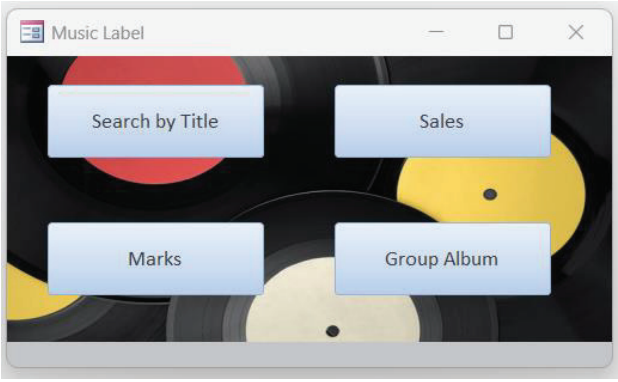


Рис. 1

Форма повинна з'являтися автоматично після відкриття бази даних.

На формі мають бути розташовані кнопки: «Search By Title», «Sales», «Marks» та «Group Album».

Після натискання на кнопку «Search By Title» мусить відкриватися форма «Search By Title», після натискання на кнопку «Sales» — форма «Sales», «Marks» — звіт «Marks», «Group Album» — звіт «Group Album».

Форма «Search By Title»

Створити форму «Search By Title», зовнішній вигляд якої має відповідати зразку.

Після відкриття користувач бачить пусту форму (*рис. 2*).

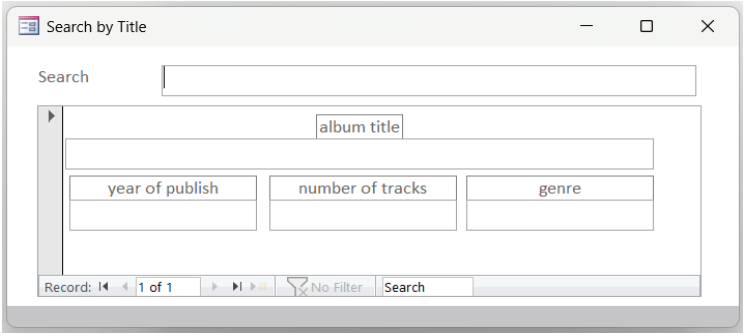


Рис. 2

Після введення у полі «Search» назви чи частини назви альбому (album_title) у формі повинна з'являтися в реальному часі інформація про альбоми, в назві яких є введені символи (*рис. 3, 4*).

Рис. 3

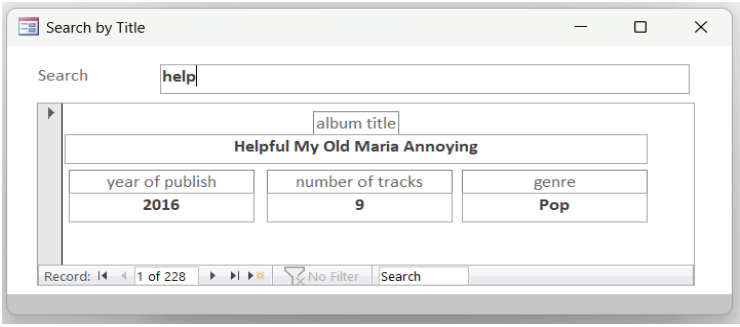


Рис. 4

Форма «Sales»

Створити форму «Sales», зовнішній вигляд якої має відповідати зразку.

Після відкриття користувач бачить пусту форму (*рис. 5*).

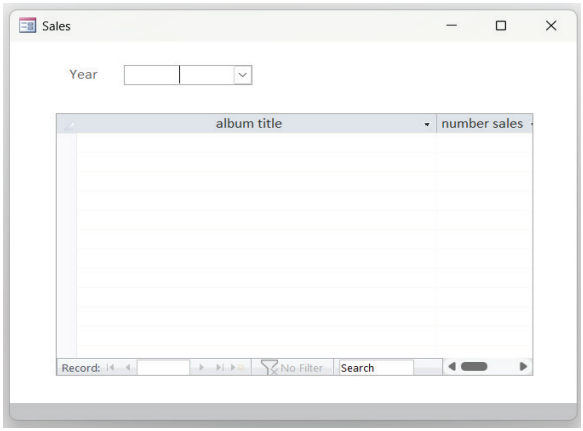


Рис. 5

На формі необхідно передбачити можливість вибору років, у які випускалися альбоми (year_of_pub), наявні у БД. Роки мають бути вказані в порядку зростання (*рис. 6*).

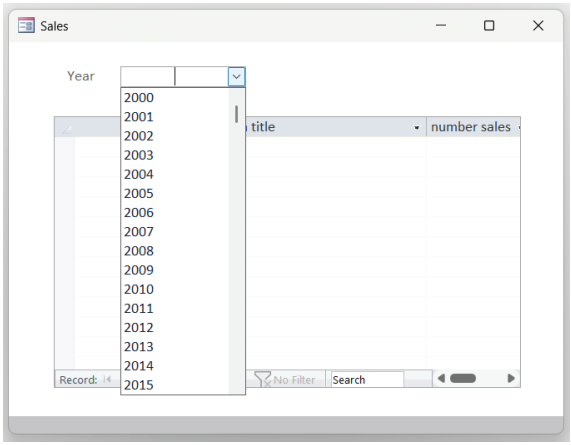


Рис. 6

Після цього у користувача мусить з'явитися можливість побачити назви альбомів, які випущені в обраному році, з 5 найбільшими та 5 найменшими кількостями продажів (num_of_sales), а також середню кількість продажів усіх альбомів цього року. Значення повинні бути впорядковані за зменшенням обсягів продажів (*рис. 7*).

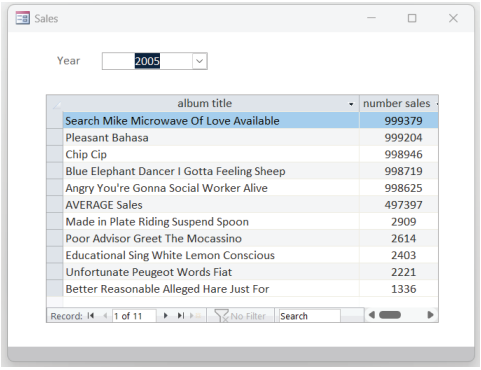


Рис. 7

Звіт «Marks»

Створити звіт «Marks», зовнішній вигляд якого має відповідати зразку (*рис. 8*).

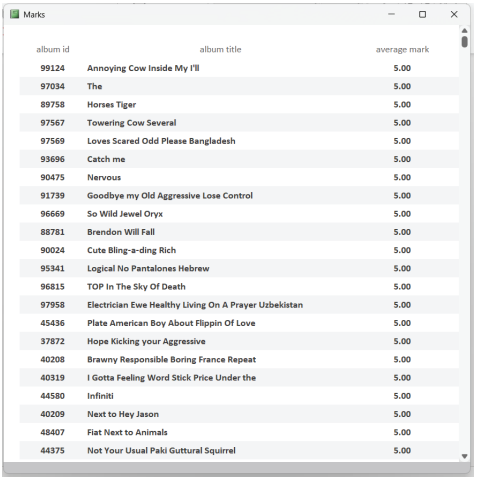


Рис. 8

У звіті повинні відображатися альбоми, у яких середня оцінка від критиків (critic, mark) більша за середню оцінку стосовно всіх альбомів, у порядку спадання середньої оцінки.

Звіт «Group Album»

Створити звіт «Group Album», зовнішній вигляд якого повинен відповідати зразку (*рис. 9).* У звіті повинні відображатися альбоми, до створення яких було залучено більше 1 артиста, а також імена артистів. Якщо артист має сценічне ім'я (art_name), то повинно відображатися воно, якщо ні – то зазначатися реальне ім'я (real_name).

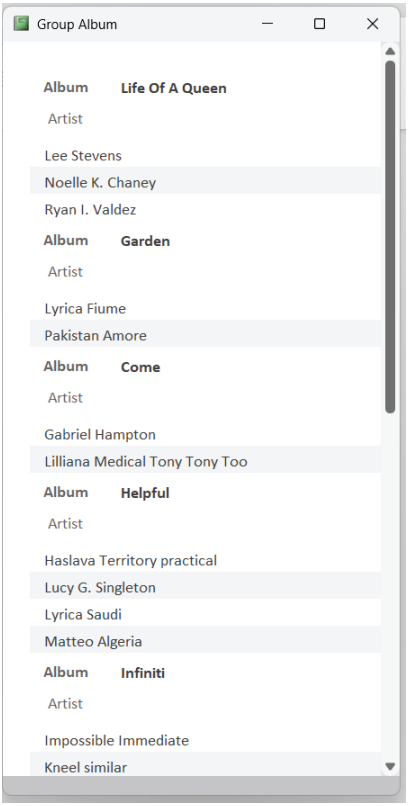


Рис. 9

Задача «Історія музики»

Завдання виконується виключно засобами MS Excel.
Результат роботи учасника потрібно зберегти у файлі «Історія музики.xlsx».

Ми слухаємо сучасну чи класичну музику, надаємо перевагу тому чи іншому виконавцю або музичному гурту, але чи замислюємося ми, як розвивалася музика як мистецтво, як змінювались музичні інструменти та стилі протягом тисячоліть? Які події в історії музики, від античності до бароко, від VII століття до н. е. до XVIII століття н. е., в різних куточках світу, змінювали уявлення людини про музику?

Учаснику необхідно в табличному процесорі згідно з інструкцією (файл «*Інструкція Excel_Історія.docx*») створити часову шкалу історії музики.

Музика цілком може бути такою ж давньою, як і саме людство, і протягом століть численні філософи намагалися пояснити наші стосунки з нею. Генрі Девід Торо одного разу сказав, що музика змушує його відчувати себе «невразливим» і безстрашним. За словами Наполеона Бонапарта: «Музика – це те, що говорить нам, що людська раса вища, ніж ми думаємо». Фрідріх Ніцше, піаніст із класичною освітою, який склав свої перші твори, коли йому було всього 18 років, вигукнув, що без музики «життя було б помилкою».

Однак мало хто продемонстрував таке ґрунтовне осмислення, як Артур Шопенгауер. Німецький мислитель народився в 1788 році на території сучасного Гданська (Польща). Він стверджував, що музика є найблагороднішим, найвеличнішим і найзначнішим з усіх видів мистецтва. Вона не тільки підноситься над іншими медіумами, такими як живопис і література, а й є єдиним засобом вираження того, що Шопенгауер вважав вищою істиною, яка править світом і всім, що є у ньому [3].

[3] Why Arthur Schopenhauer thought music was the greatest of all artforms. *Big Think*. 2021. October 26. URL: https://bigthink.com/high-culture/schopenhauer-music-will/ (дата звернення: 25.02.2025).

Інструкція

Файл «Історія музики.xlsx».

Аркуш «history_music».

На аркуші розміщено інформацію про найважливіші події в історії музики від античності до бароко, від VII століття до н. е. до XVIII століття н. е.

Аркуш «timeline»

Створити часову шкалу історії музики (див. *рис. 1*).

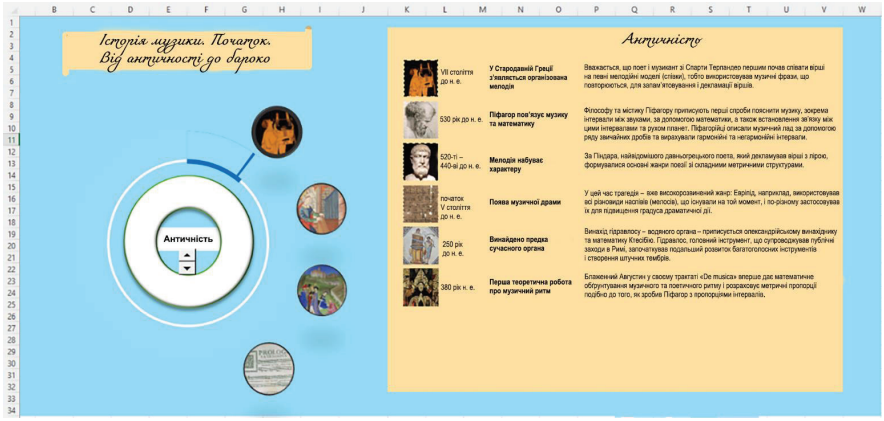
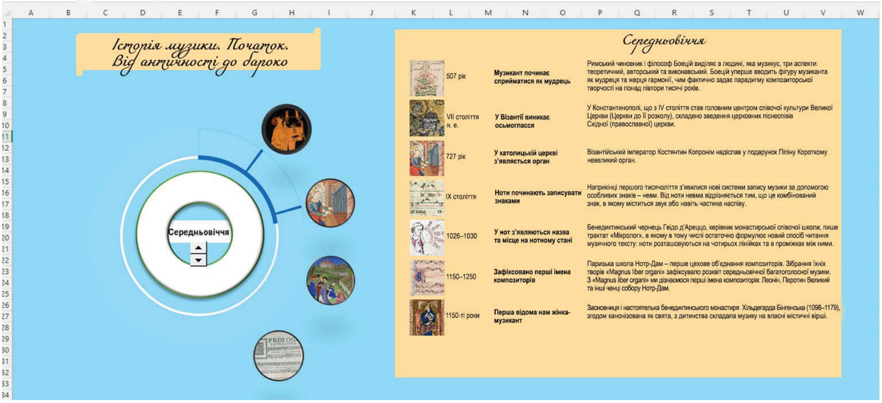
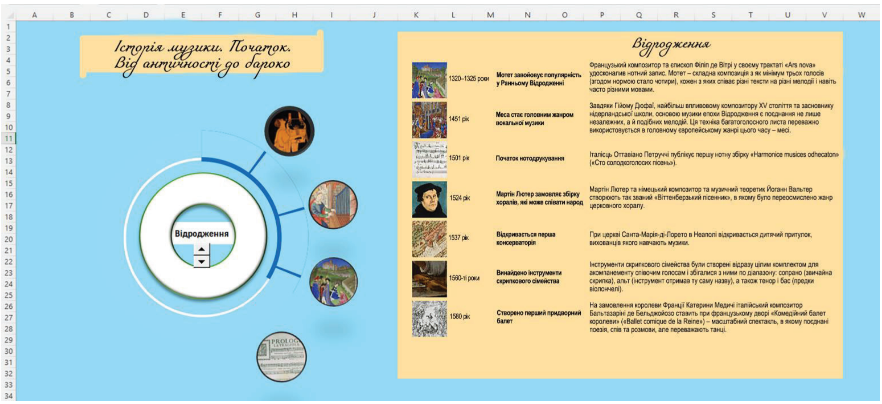


Рис. 1

Лічильник вказує на історичний період, відповідна частина кола змінює колір, праворуч з'являється інформація про видатні події в історії музики у певний історичний період (*рис. 2–4*).


Історія музики. Початок.
Від античності до бароко
Середньовіччя
Середньовіччя
507 рік
Музикант починає сприйматися як мудрець
Римський чиновник і філософ Боецій виділяє в людині, яка музикує, три аспекти: теоретичний, авторський та виконавський. Боецій уперше вводить фігуру музиканта як мудреця та жерця гармонії, чим фактично задає парадигму композиторської творчості на понад півтори тисячі років.
VII століття н. е.
У Візантії виникає осьмогласся
У Константинополі, що з IV століття став головним центром співочої культури Великої Церкви (Церкви до її розколу), складено зведення церковних піснеспівів Східної (православної) церкви.
727 рік
У католицькій церкві з'являється орган
Візантійський імператор Костянтин Копронім надіслав у подарунок Піпіну Короткому невеликий орган.
IX століття
Ноти починають записувати знаками
Наприкінці першого тисячоліття з'явилися нові системи запису музики за допомогою особливих знаків – невм. Від ноти невма відрізняється тим, що це комбінований знак, в якому міститься звук або навіть частина наспіву.
1026–1030
У нот з'являються назва та місце на нотному стані
Бенедиктинський чернець Гвідо д'Ареццо, керівник монастирської співочої школи, пише трактат «Мікролог», в якому в тому числі остаточно формулює новий спосіб читання музичного тексту: ноти розташовуються на чотирьох лінійках та в проміжках між ними.
1150–1250
Зафіксовано перші імена композиторів
Паризька школа Нотр-Дам – перше цехове об'єднання композиторів. Зібрання їхніх творів «Magnus liber organi» зафіксувало розквіт середньовічної багатоголосної музики. З «Magnus liber organi» ми дізнаємося перші імена композиторів: Леонін, Перотин Великий та інші ченці собору Нотр-Дам.
1150-ті роки
Перша відома нам жінка-музикант
Засновниця і настоятелька бенедиктинського монастиря Хільдегарда Бінгенська (1098–1179), згодом канонізована як свята, з дитинства складала музику на власні містичні вірші.


Історія музики. Початок.
Від античності до бароко
Відродження
Відродження
1320–1325 роки
Мотет завойовує популярність у Ранньому Відродженні
Французький композитор та єпископ Філіп де Вітрі у своєму трактаті «Ars nova» удосконалив нотний запис. Мотет – складна композиція з як мінімум трьох голосів (згодом нормою стало чотири), кожен з яких співає різні тексти на різні мелодії і навіть часто різними мовами.
1451 рік
Меса стає головним жанром вокальної музики
Завдяки Гійому Дюфаї, найбільш впливовому композитору XV століття та засновнику нідерландської школи, основою музики епохи Відродження є поєднання не лише незалежних, а й подібних мелодій. Ця техніка багатоголосного листа переважно використовується в головному європейському жанрі цього часу – месі.
1501 рік
Початок нотодрукування
Італієць Оттавіано Петруччі публікує першу нотну збірку «Harmonice musices odhecaton» («Сто солодкоголосих пісень»).
1524 рік
Мартін Лютер замовляє збірку хоралів, які може співати народ
Мартін Лютер та німецький композитор та музичний теоретик Йоганн Вальтер створюють так званий «Віттенберзький пісенник», в якому було переосмислено жанр церковного хоралу.
1537 рік
Відкривається перша консерваторія
При церкві Санта-Марія-ді-Лорето в Неаполі відкривається дитячий притулок, вихованців якого навчають музики.
1560-ті роки
Винайдено інструменти скрипкового сімейства
Інструменти скрипкового сімейства були створені відразу цілим комплектом для акомпанементу співочим голосам і збігалися з ними по діапазону: сопрано (звичайна скрипка), альт (інструмент отримав ту саму назву), а також тенор і бас (предки віолончелі).
1580 рік
Створено перший придворний балет
На замовлення королеви Франції Катерини Медичі італійський композитор Бальтазаріні де Бельджойозо ставить при французькому дворі «Комедійний балет королеви» («Ballet comique de la Reine») – масштабний спектакль, в якому поєднані поезія, спів та розмови, але переважають танці.

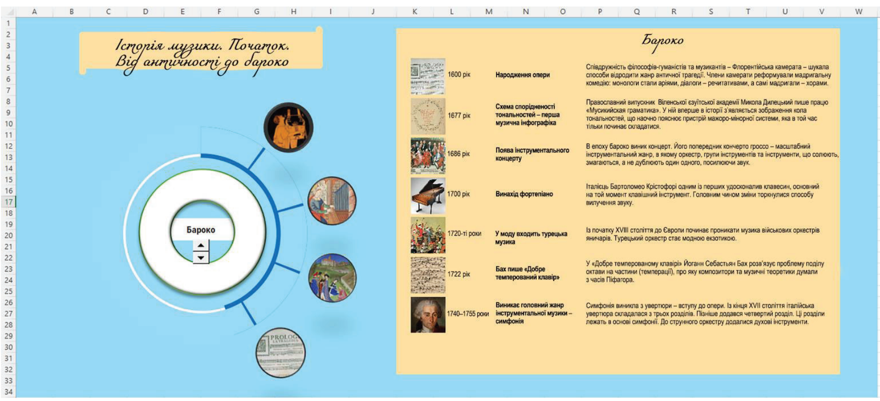

Історія музики. Початок.
Від античності до бароко
Бароко
Бароко
1600 рік
Народження опери
Співдружність філософів-гуманістів та музикантів – Флорентійська камерата – шукала способи відродити жанр античної трагедії. Члени камерати реформували мадригальну комедію: монологи стали аріями, діалоги – речитативами, а самі мадригали – хорами.
1677 рік
Схема спорідненості тональностей – перша музична інфографіка
Православний випускник Віленської єзуїтської академії Микола Дилецький пише працю «Мусикийская граматика». У ній вперше в історії з'являється зображення кола тональностей, що наочно пояснює пристрій мажоро-мінорної системи, яка в той час тільки починає складатися.
1686 рік
Поява інструментального концерту
В епоху бароко виник концерт. Його попередник кончерто гроссо – масштабний інструментальний жанр, в якому оркестр, групи інструментів та інструменти, що солюють, змагаються, а не дублюють один одного, посилюючи звук.
1700 рік
Винахід фортепіано
Італієць Бартоломео Крістофорі одним із перших удосконалив клавесин, основний на той момент клавішний інструмент. Головним чином зміни торкнулися способу вилучення звуку.
1720-ті роки
У моду входить турецька музика
Із початку XVIII століття до Європи починає проникати музика військових оркестрів яничарів. Турецький оркестр стає модною екзотикою.
1722 рік
Бах пише «Добре темперований клавір»
У «Добре темперованому клавірі» Йоганн Себастьян Бах розв'язує проблему поділу октави на частини (темперації), про яку композитори та музичні теоретики думали з часів Піфагора.
1740–1755 роки
Виникає головний жанр інструментальної музики – симфонія
Симфонія виникла з увертюри – вступу до опери. Із кінця XVII століття італійська увертюра складалася з трьох розділів. Пізніше додався четвертий розділ. Ці розділи лежать в основі симфонії. До струнного оркестру додалися духові інструменти.

Рис. 2–4

Задача «Музичний автомат»

Завдання виконується виключно засобами MS Excel.
Результат роботи учасника потрібно зберегти у файлі «Музичний автомат.xlsx».

З розвитком технологій для виконання музики у місцях громадського відпочинку почали використовуватися пристрої, призначені для автоматизованого програвання музичних композицій, — музичні автомати. За невелику плату відвідувач міг самостійно ввімкнути улюблену мелодію або пісню, вкинувши в монетоприймач декілька монет. Часто це можна побачити у старих фільмах про життя у XIX–XX сторіччях. Зі зміною технологій змінювалися й музичні автомати. Проте, хоча зараз їх аналогом є музичні інтернет-сервіси, завжди цікаво зазирнути за лаштунки історії, щоб подивитись, як жили попередні покоління людства.

Учасник має створити віртуальний музичний автомат згідно з інструкцією *(файл «Інструкція Excel.docx»)*.

Інструкція

Файл «Музичний автомат.xlsx».

Аркуш «Data».

У задачі використовується фрагмент набору даних «Набір музичних даних Spotify» [4].

Описові характеристики пісні містять таку інформацію:

- Id — код композиції (пісні);
- playlist_genre — жанр;
- track_artist — виконавець;
- track_album_name — альбом;
- track_name — назва пісні;
- duration_ms — тривалість в мілісекундах;
- cover — обкладинка альбому;
- playing price — ціна прослуховування.

[4] Ameh S. Spotify Music Dataset. Popular and Not so popular songs. URL: https://www.kaggle.com/datasets/solomonameh/spotify-music-dataset/data (дата звернення: 25.02.2025).

Аркуш «cover»

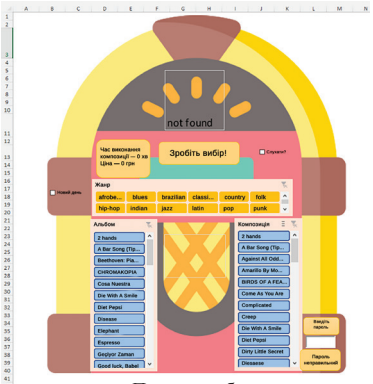
На аркуші міститься зображення обкладинки кожного альбому в збільшеному вигляді. Зображення вбудовані в комірки.

Аркуш «img»

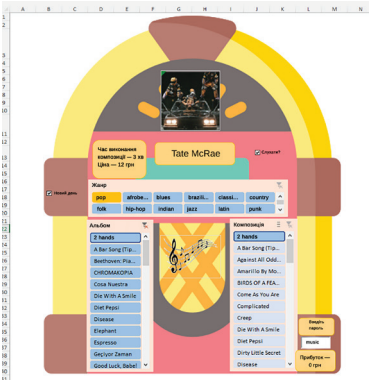
На аркуші зображено малюнки, за допомогою яких можна створити музичний автомат.

Аркуш «JukeBox»

На аркуші учасник повинен розмістити віртуальний музичний автомат (*рис. 1*). Дані для його створення знаходяться на аркушах «Data», «cover», «img».



Початок роботи



Грає вибрана композиція

Завершення відтворення пісні

Завершення відтворення другої пісні

Рис. 1

Елементи «JukeBox»

1. Слайсер «Жанр». Дані зі стовпця «playlist_genre» даних, аркуш «Data».

Формат слайсера зображено на рис. 2. Мультивибору немає.

Рис. 2

2. Слайсер «Альбом». Дані зі стовпця «track_album_name» даних, аркуш «Data». Формат слайсера зображено на рис. 3. Мультивибору немає.

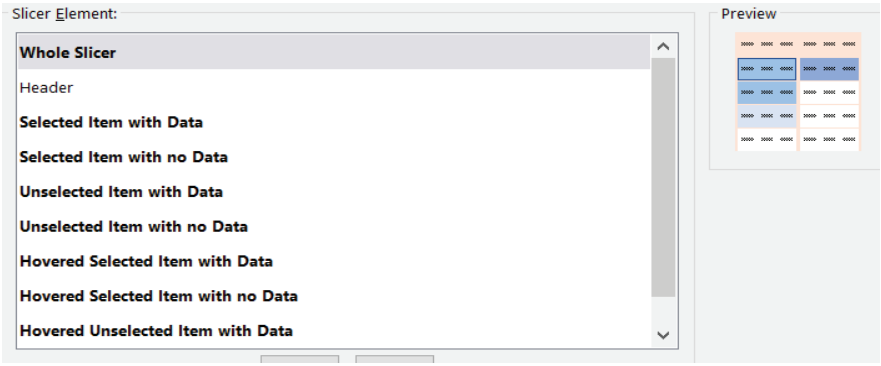


Рис. 3

3. Слайсер «Композиція».

Дані зі стовпця «track_name» даних, аркуш «Data».
Формат слайсера зображено на рис. 3. Мультивибору немає.

4. Обкладинка альбому вибраної пісні.

Інформація про час виконання у хвилинах і вартість прослуховування в грн (*рис. 4*).

Рис. 4

5. Прапорці «Новий день», «Слухати?»

6. Напис «Введіть пароль», текстове поле, напис «Прибуток S грн», що містить інформацію про загальну суму, накопичену в автоматі у результаті прослуховування пісень при введенні в поле пароля «music», або повідомлення «Пароль неправильний» в іншому випадку (*рис. 5*).

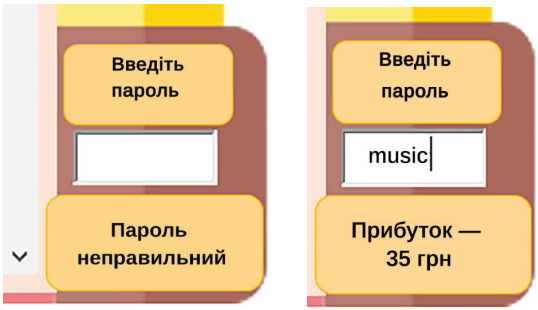


Рис. 5

Робота «JukeBox»

У процесі роботи автомат має обчислювати загальну суму, що назбиралася за прослуховування пісень. Якщо прапорець «Новий день» вимикається, попередня сума анулюється. Після встановлення прапорця починається накопичення нової суми.

Початок роботи (*рис. 1*):

1) встановити прапорець «Новий день»;

2) фільтри на слайсерах вимкнуті;

3) пароль знятий.

Робота:

1. Вибрати жанр, чи альбом, чи композицію. У слайсері «Альбоми» виділяються альбоми цього жанру, в слайсері «Композиція» – пісні (*рис. 6*). З'являється обкладинка першого альбому зі слайсера «Альбом», інформація про виконавця, час прослуховування і вартість прослуховування першої пісні першого альбому. Маємо можливість вибрати інший альбом чи жанр.

Рис. 6

2. Якщо вибір зроблено, встановлюємо прапорець «Слухати?». З динаміка лунає музика у вигляді віртуальної нотної хвилі (*рис. 7*).

Рис. 7

3. Знімаємо прапорець «Слухати?».
Якщо введемо пароль в текстове поле, побачимо накопичену суму (*рис. 8*).

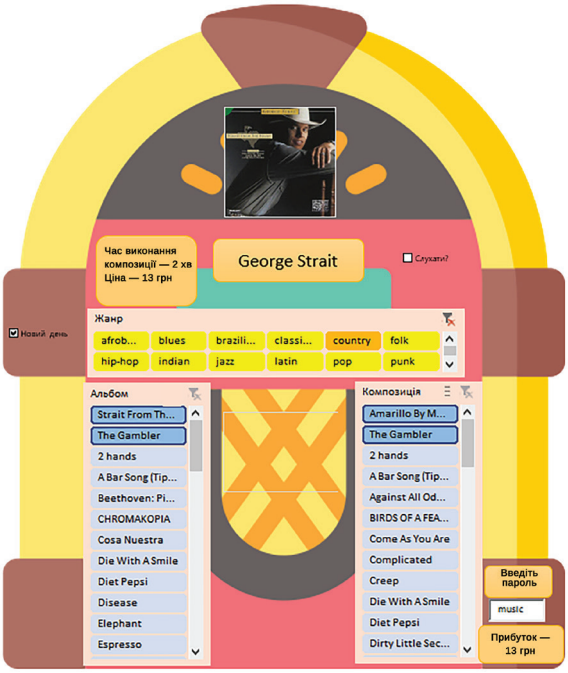


Рис. 8

Вибираємо нову композицію, працюємо за наведеними вище правилами використання музичного автомата. Наприклад, вибрали іншу композицію, встановили «Слухати?», потім зняли прапорець (бо прослухали), вартість прослуховування додалася до попередньої (*рис. 9*).

Рис. 9

Задача «Дискографія»

Завдання виконується виключно засобами MS Word.
Результат роботи учасника потрібно зберегти у файлі «oe_discography.docx».

Будь-який музичний гурт, який піднісся на олімп популярності й залишився в історії музики, був визнаний критиками та шанувальниками, відзначений музичними преміями та завойовував престижні музичні нагороди, має багато виданих альбомів пісень, що стали популярними.

«Океан Ельзи» — один із найвідоміших українських гуртів, що неодноразово був визнаний критиками й шанувальниками найкращим рок- і live-гуртом Східної Європи. Цей гурт збирав стадіони шанувальників у всьому світі й випустив не один альбом музичних хітів.

Учаснику необхідно відповідно до інструкції суворо за зразком (файл «*Інструкція Word.docx»*) відтворити інтерактивний довідник творчості гурту «Океан Ельзи».

Інструкція

Учасник має наповнити файл «*oe_discography.docx»,* який повинен містити історію та біографію відомого українського музичного гурту «Океан Ельзи». Файл складається з назви гурту, емблеми гурту, фото гурту, тексту з історією гурту та переліку випущених альбомів. Форматування мусить бути виконано суворо за зразком.

Після назви кожного альбому необхідно додати елемент «Прапорець». При виділенні всього документа та натисканні F9 в документі повинна відобразитись інформація про кожен альбом, для якого увімкнений прапорець. Інформація про альбоми, для яких не увімкнений прапорець, відображатись не повинна.

O.E.

Океан Ельзи

біографія і творчість

Жанр: *інді-рок, інді-поп, арт-рок*
Дата створення: *1994*
Місто: *Львів*
Лейбл: *Moon Records, EMI*

"Океан Ельзи" – один з найвідоміших українських гуртів, що неодноразово визнавався критиками і шанувальниками найкращим рок- і live-гуртом Східної Європи.

Гурт зародився у Львові 12 жовтня 1994 року. Зараз учасниками культового бенду є Святослав Вакарчук (вокал), Денис Глінін (ударні), Денис Дудко (бас-гітара) Мілош Єліч (клавішні) та Владімір Опсеніца (гітара). З часу заснування склад "океанів" неодноразово змінювався і лише Святослав Вакарчук залишається його незмінним лідером і за сумісництвом автором більшості пісень.

Практично щорічно гурт був номінантом різноманітних вітчизняних премій і неодноразово завойовував престижні музичні нагороди: переміг в трьох номінаціях експертної премії ("Кращий рок-гурт "Виконавці року"); у чотири статуетки у квітні 2007-го став музичної премії "FUZZ" простору; у 2012-му як "Найкращий гурт му Вакарчук і Ко "YUNA" аж в 4 номінаціях (фронтмен переміг у номінаціях "Найкращий композитор" і "Найкращий автор слів", а ОЕ – "Найкращий гурт" і "Альбом року").

"ShowBizAwards 2005" року", "Альбом року", січні 2006-го отримав "Ukrainian Rock Awards"; володарем експертної як кращий рок-гурт СНГ- отримав премію "YUNA" двадцятиріччя"; у 2014- заволоділи премією

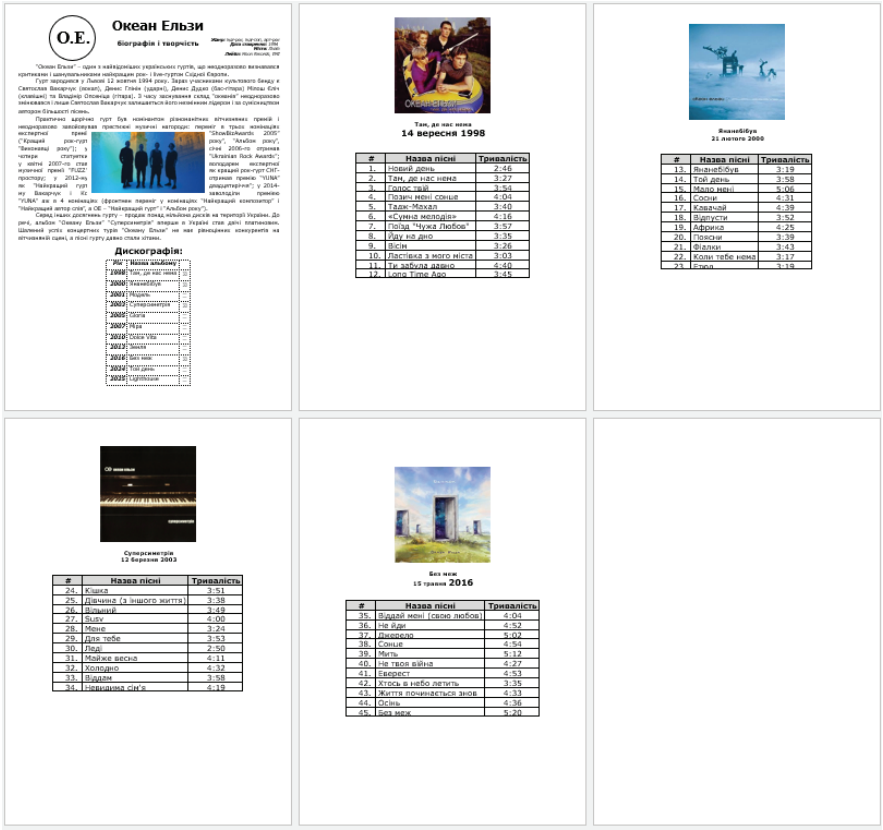
Серед інших досягнень гурту – продаж понад мільйона дисків на території України. До речі, альбом "Океану Ельзи" "Суперсиметрія" вперше в Україні став двічі платиновим. Шалений успіх концертних турів "Океану Ельзи" не має рівноцінних конкурентів на вітчизняній сцені, а пісні гурту давно стали хітами.

Дискографія:

Рік	**Назва альбому**	
1998	Там, де нас нема	☐
2000	Янанебібув	☐
2001	Модель	☐
2003	Суперсиметрія	☐
2005	Gloria	☐
2007	Міра	☐
2010	Dolce Vita	☐
2013	Земля	☐
2016	Без меж	☐
2024	Той день	☐
2025	Lighthouse	☐

Вигляд першої сторінки файлу «oe_discography.docx»

Інформація про альбом починається з нової сторінки, містить його зображення, назву, дату випуску та перелік треків.



Вигляд всього файлу «oe_discography.docx» з обраними прапорцями

Оформлення повинно бути зроблено суворо за зразком.

Учаснику необхідно здати папку з усіма матеріалами та файлами із виконаними завданнями.

Примітка. У деяких випадках при правильно виконаному завданні після виділення всього тексту та оновлення даних натисканням F9 може з'явитися повідомлення про помилку: «Помилка! Указано неправильне ім'я файлу».

Якщо таке відбулося, необхідно заново зберегти файл у відповідну папку за допомогою пункту «Зберегти як…» у меню «Файл».

Задача «Гітара»

Завдання виконується виключно засобами MS PowerPoint.
Результат роботи учасника потрібно зберегти у файлі «Гітара.pptx».

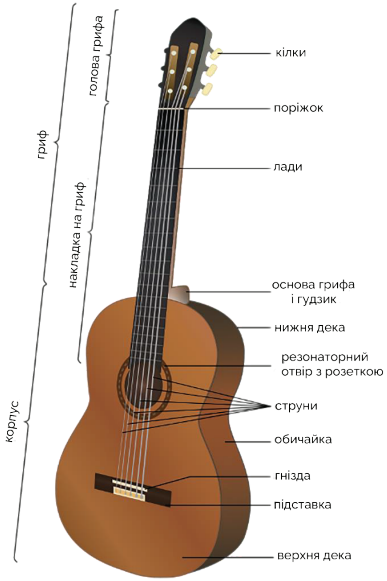
Музичним супутником у житті чи не кожної людини є гітара. Цей інструмент популярний як серед знавців музики, так і з-поміж людей, які люблять музику і пісні, діляться своїми здібностями й талантами з рідними і друзями, покращуючи настрій музичним супроводом співу в компаніях.

Кожен може спробувати навчитися грати на гітарі професійно чи опанувати ці навички самостійно, так би мовити, для себе, щоб стати душею компанії на відпочинку, в поході чи подорожі.

Проте ж як прості, так і складні мелодії, якими нас дивує цей інструмент, з'являються внаслідок фізичних процесів коливання струни, що їх створюють звукові хвилі.

Учаснику необхідно суворо за зразком (файл *«Гітара_зразок.mp4»*) та відповідно до інструкції (файл «*Інструкція PowerPoint.docx»*) зробити керовану презентацію процесу створення звукових хвиль за рахунок надання медіатором вібрації струнам гітари.

Гіта́ра (ісп. guitarra) — струнний музичний інструмент родини лютень. Звук утворюється завдяки вібрації струн і підсилюється за допомогою резонатора — деки інструмента.



Гітари поділяються на класичні, акустичні, електроакустичні, електричні та напівакустичні. Крім того, гітари розрізняють за кількістю та матеріалом виготовлення струн, будовою корпусу і призначенням. Гітару використовують як сольний, акомпанувальний та ансамблевий інструмент у різних музичних напрямах.

Т. Г. Шевченко Портрет невідомого з гітарою (1848—1849)

Сама назва «гітара», ймовірно, походить від «кіфари» — давньогрецького щипкового інструмента (κιθάρα), схожого на ліру. Пізніше грецьке слово потрапило до латинської, арабської, іспанської мов, звідки вже було запозичене іншими європейськими мовами, в тому числі й українською [5].

Гітара досі є найпоширенішим музичним інструментом. І можна побачити її зображення на картинах художників різних епох.

Кожен музичний інструмент створює звуки, які складаються в мелодію. У створенні звуків на гітарі велику роль відіграють основні прийоми звуковидобування – щипок та удар. Саме вони приводять до коливання струни і, відповідно, створення звукової хвилі. Музичні звуки характеризуються висотою, силою, тембром, тривалістю. Висота звука визначається частотою коливань натягнутої струни, а сила звука — інтенсивністю коливання струни, яка залежить від амплітуди коливання.

Інструкція

Учасник має суворо за зразком (файл *«Гітара _зразок.mp4»*) відтворити модель гітари та зробити модель процесу відтворення звуків в анімації струн і відображенні їхніх коливань на екрані осцилографа.

Якщо натиснути на маркер струни (файл *«Гітара _зразок.mp4»*), має відтворитись ефект її коливання.

Після натиснення на червону кнопку ввімкнення осцилографа на його екрані має з’явитися зображення сітки, а під час натискання на

[5] Гітара. URL: https://uk.wikipedia.org/wiki/Гітара (дата звернення: 25.02.2025).

маркери струн на ньому мають відображатись синусоїди – графіки звукових хвиль.

Швидке натиснення на маркери декількох струн спричиняє одночасне відтворення на екрані осцилографа відповідних хвиль.

Анімація має починати відтворюватись автоматично з моменту запуску слайда напоказ.

Розв'язання завдання має відбуватись у наданому авторами файлі *«Гітара.pptx»,* у ньому ж повинен зберігатися розв'язок.

ІІ тур

Задача «MusicLabelData2»

Завдання виконується виключно засобами MS Access.
Результат роботи учасника потрібно зберегти у файл «MusicLabelData2.accdb».

У базі даних (файл «*MusicLabelData2.accdb*») міститься інформація, необхідна музичній компанії для формування аналітики стосовно наявних музичних альбомів, виконавців у різних жанрах.

Учаснику необхідно у базі даних (файл «*MusicLabelData2.accdb*») створити запити, суворо дотримуючись інструкцій (файл *«Інструкція Access.docx»*).

Інструкція

Учаснику відповідно до наведених нижче настанов необхідно у базі даних *«MusicLabelData2.accdb»*, яка містить синтетично згенерований набір музичних даних для невеликої уявної музичної компанії, створити запити, що повертають:

- Query 1: справжні імена 10 виконавців (real_name), які є авторами найбільшої кількості треків (num_of_tracks), впорядкованих за спаданням кількості;
- Query 2: назви 3 альбомів (album_title), які найчастіше зустрічаються у БД в порядку спадання значень;
- Query 3: назви альбомів (album_title), автором яких є артист, чиє реальне ім’я (real_name) або його частина введені користувачем з клавіатури;
- Query 4: назви 3 жанрів (genre) з найбільшою кількістю та 3 жанрів з найменшою кількістю альбомів, які випущені у році, введеному користувачем з клавіатури, в порядку спадання кількості;
- Query 5: назви 5 жанрів (genre) з найбільшою середньою оцінкою (mark) альбомів цього жанру за версією рейтингу критиків

(critic_name), введені користувачем з клавіатури та впорядковані за спаданням;

- Query 6: назви 3 найбільш рідкісних спеціальностей (roles) артистів, вказані у порядку спадання рідкості.

Під час виконання завдання заборонено змінювати типи полів та додавати інші поля у таблицях БД, додавати інші таблиці у БД [6].

Задача «Оперний театр»

Завдання виконується виключно засобами MS Excel.
Результат роботи учасника потрібно зберегти у файл «Оперний_театр.xlsx».

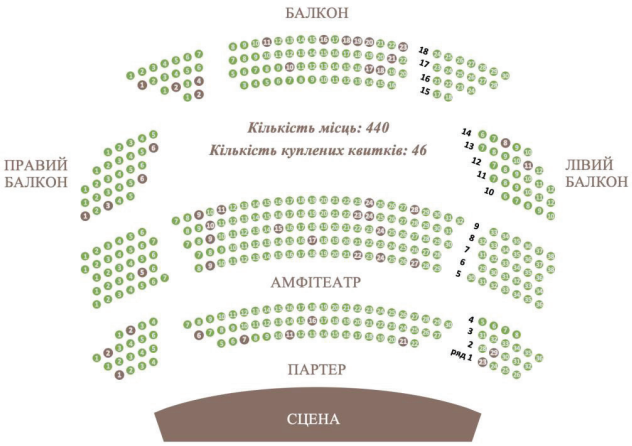
Оперні театри є центрами музичної культури, у яких глядачі насолоджуються виставами світового рівня. Кожен театр має схему розташування місць, і під час продажу квитків важливо відображати на ній зайняті та вільні місця.

Учаснику пропонується, суворо дотримуючись інструкції (файл *«Інструкція_Оперний_театр.docx»*), створити у табличному процесорі систему, за допомогою якої можна вносити в базу інформацію про куплені квитки та відображати схему театру з позначенням зайнятих місць.

Інструкція

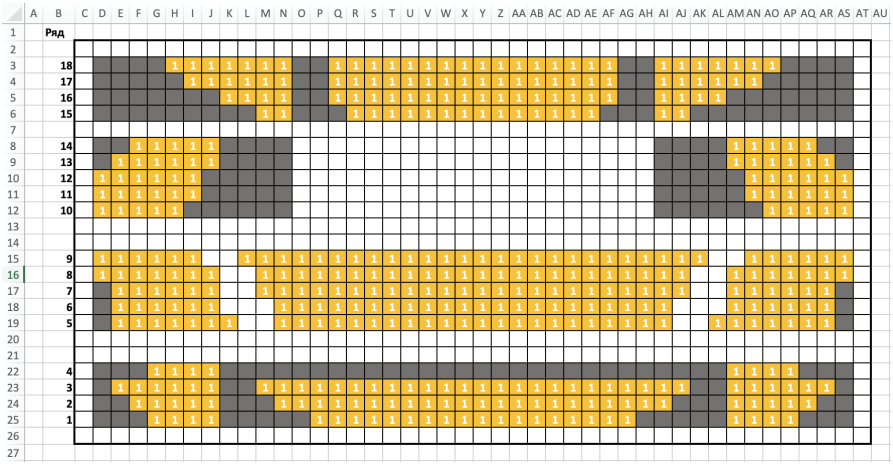
У межах цього завдання пропонуємо вам поринути в атмосферу театру, вистав і музики та розробити систему купівлі квитків на оперу.

[6] Дані взято з Kaggle. URL: https://www.kaggle.com/datasets/revilrosa/ music-label-dataset/data (дата звернення: 25.02.2025).



Вхідні дані

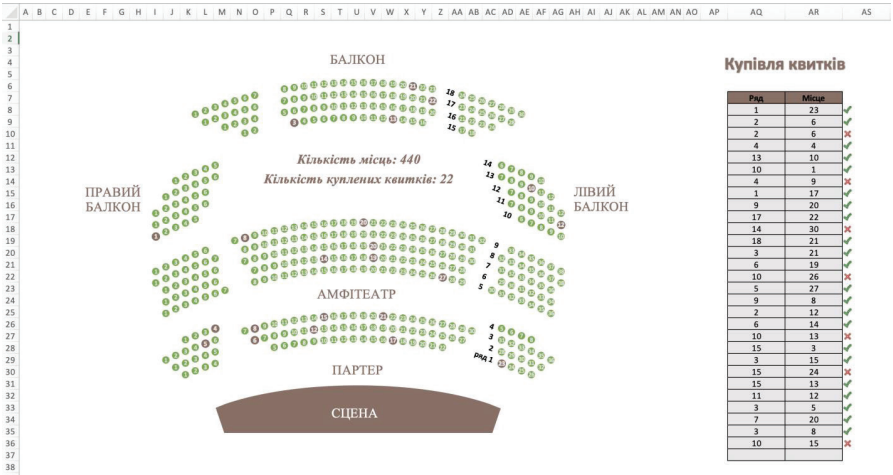
На аркуші «Схема» у діапазоні C2:AT26 розташована схема оперного театру, що містить 18 рядів. Ряди 1–4 позначають партер, 5–9 — амфітеатр, 10–14 — лівий та правий балкони, а 15–18 — основний балкон. Схема театру може змінюватися, але лише в межах відповідних сірих зон. Доступні для сидіння місця позначаються «1» та автоматично підсвічуються жовтим кольором.



Під час перевірки журі може змінювати схему зали в межах сірих клітинок.

Завдання

На аркуші «Квитки» необхідно реалізувати суворо за зразком наступну систему (приклад роботи показано у відео *«Робота_системи.mp4»*):



Ряд	Місце
1	23
2	6
2	6
4	4
13	10
10	1
4	9
1	17
9	20
17	22
14	30
18	21
3	21
6	19
10	26
5	27
9	8
2	12
6	14
10	13
15	3
3	15
15	24
15	13
11	12
3	5
7	20
3	8
10	15

Купівля квитків

У стовпцях AQ та AR (починаючи з рядка 8) вписуються ряд та місце відповідно. Праворуч від них розташовані позначення: зелена галочка, за умови, що місце існує та ще не було зайняте, та червоний хрестик — у іншому випадку.

За умови, що квиток на місце можна прибрати, це місце зафарбовується коричневим кольором на інтерактивній мапі зали зліва.

Список «Купівля квитків» на аркуші «Квитки» завжди містить один вільний рядок для вписування ряду та місця. При купівлі одного квитка вільне місце в списку з'являється на наступному після нього рядку.

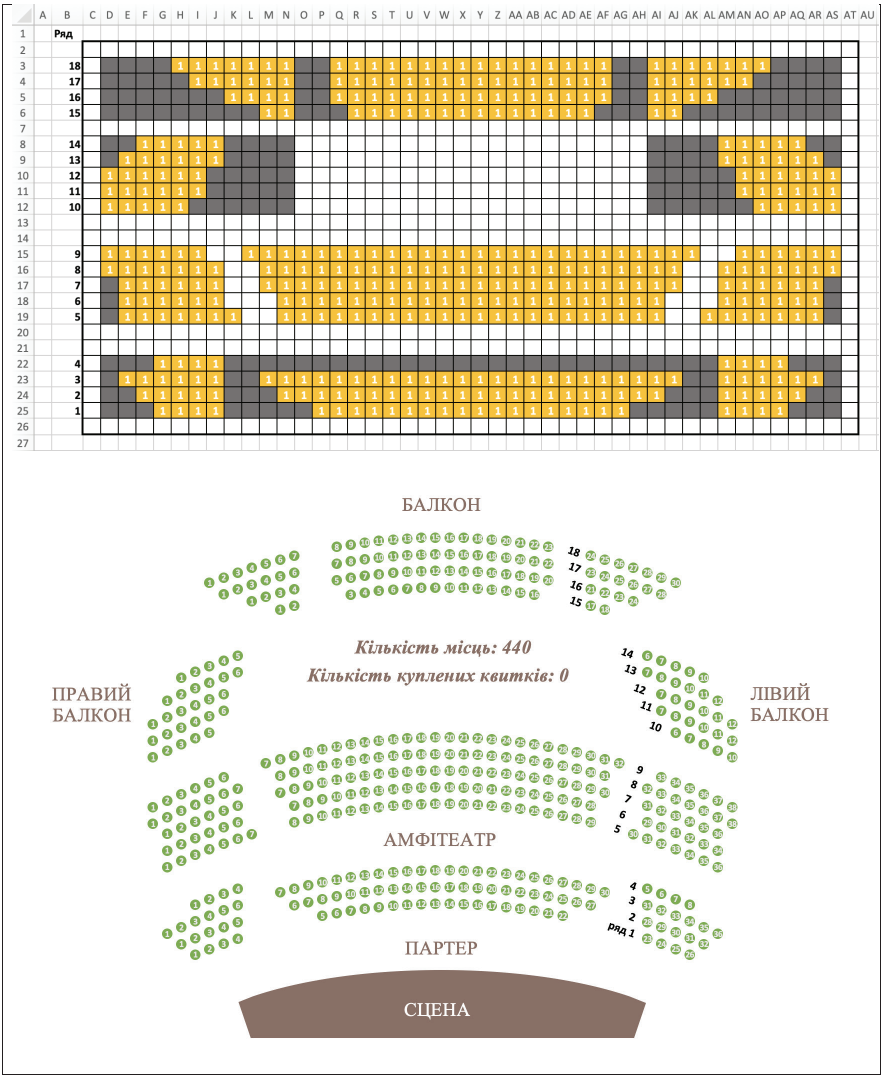
Приклад

Примітка. На третій картинці біля другого запису (ряд 1 місце 30) стоїть червоний хрестик, адже такого місця не існує на схемі, а біля четвертого запису (ряд 2 місце 15) стоїть червоний хрестик, бо це місце вже зайняте (було куплене в третьому записі).

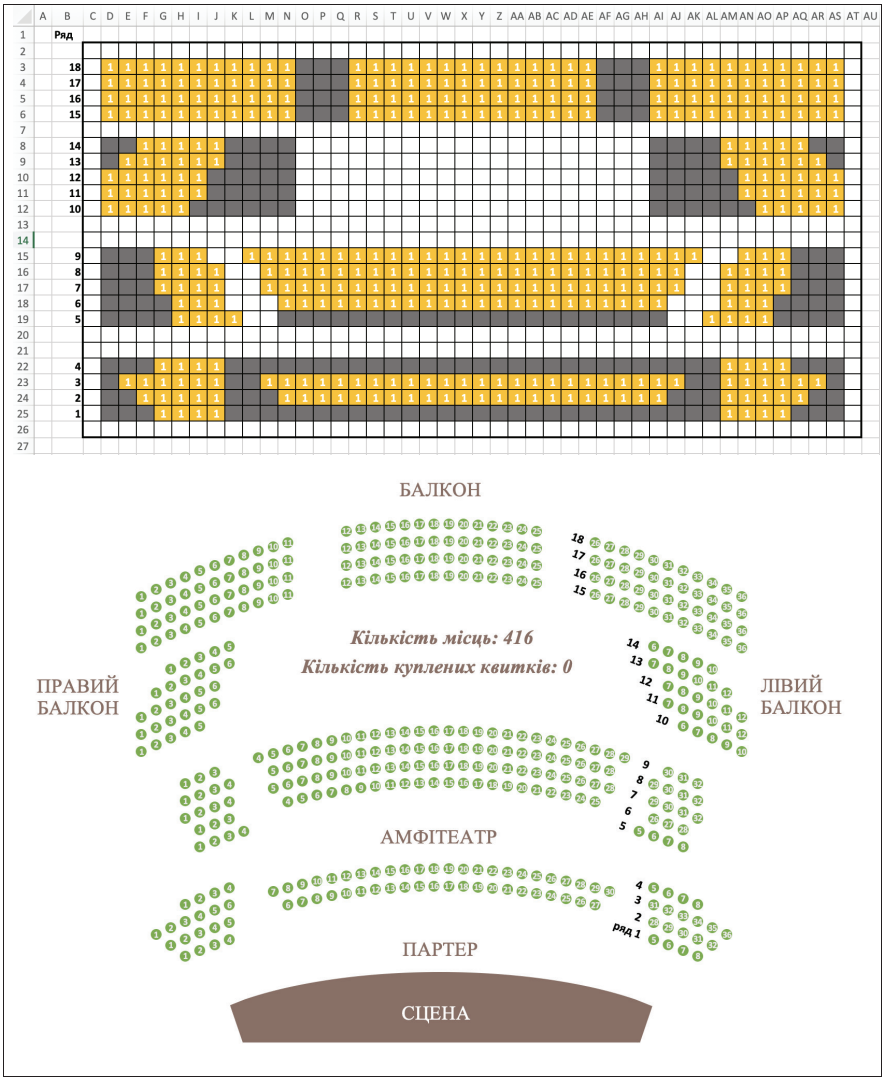
Інтерактивна мапа зали

Відображення мапи зали на аркуші «Купівля квитків» повинно відповідати аркушу «Схема». Нумерація місць у залі й на інтерактивній мапі зали повинна бути однаковою, а ряди — відображатися вигнутими. Приклади наведені далі.

Примітка. Ряди зали описують параболу, загальна формула якої $y = x^2$. Журі під час перевірки допускають незначне відхилення форми вигину рядів від зразка.


Ряд
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
БАЛКОН
Кількість місць: 440
Кількість куплених квитків: 0
ПРАВИЙ БАЛКОН
ЛІВИЙ БАЛКОН
АМФІТЕАТР
ПАРТЕР
РЯД 1
СЦЕНА

Приклад 1



Приклад 2

На мапі зали підписані сектори, згадані вище – балкон (ряди 15–18), правий і лівий балкони (ряди 10–14), амфітеатр (ряди 5–9), партер (ряди 1–4), а також сцена, розташована внизу мапи.

У центрі на мапі розташована статистична інформація, яка відображає загальну кількість місць (місця, що наявні на схемі) та кількість куплених квитків (місця, позначені зеленою галочкою у списку «Купівля квитків»).

Технічні умови:

- інтерфейс має відповідати наданим зразкам;
- документ має містити аркуші «Схема» та «Купівля квитків»;
- на аркуші «Схема» задається схема зали оперного театру в межах сірих зон;
- на аркуші «Купівля квитків» відображається список квитків для купівлі (справа) та інтерактивна мапа зали (зліва), що будується на основі схеми;
- при додаванні квитка у список «Купівля квитків» можливість його придбати має відображатися зеленими галочками та червоними хрестиками;
- додані до списку місця, на які можливо придбати квитки, відображаються коричневим кольором на мапі, як зайняті;
- ряди на інтерактивній мапі зали відображаються вигнутими у формі параболи;
- місця на мапі зали відображаються послідовно, зліва направо; нумерація місць у залі й на інтерактивній мапі зали повинна бути однаковою;
- статистична інформація знаходиться в центрі мапи та відображає загальну кількість місць у залі й кількість куплених квитків;
- відеозразок роботи системи знаходиться у файлі *«Робота_системи.mp4»*;
- результат виконання завдання потрібно зберегти у файлі *«Оперний_театр.xlsx»;*
- використання VBA заборонено.

Задача «Європейський музичний олімп»

Завдання виконується виключно засобами MS Excel.

Результат роботи учасника потрібно зберегти у файлі «EuropeanTop.xlsx».

У XX столітті одним із найбільш поширених носіїв звукових даних була вінілова платівка. Саме вона стала символом популярності музичних творів і виконавців. Деякі платівки розходилися мільйонними тиражами. Одне було незмінним – лаконічний дизайн вінілової платівки.

Учаснику пропонується з використанням можливостей виключно табличного процесора, суворо дотримуючись інструкції (файл *«Інструкція Excel_Європейський музичний олімп.docx»*), створити засіб для конструювання дизайну вінілової платівки.

Вінілові платівки були одним із найбільш поширених носіїв музики у XX столітті. Їхня історія зароджувалася на початку 1900-х років, коли музика записувалася на круглі платівки з шелаку (твердої смоли), які оберталися зі швидкістю 78 обертів за хвилину. Ці платівки були крихкими, важкими і могли містити лише кілька хвилин звуку на кожній стороні, але згодом їх почали виготовляти з вінілового пластику, та й час відтворюваної музики збільшився. У 1950–1980-х роках вінілові платівки стали основним носієм музики — їх використовували для запису альбомів, синглів, класичної музики та навіть аудіокниг.

Платівки стали важливою частиною музичної культури, а їхні обкладинки перетворилися на мистецтво.

З появою компакт-дисків (CD) популярність вінілових платівок почала зменшуватися, адже CD пропонували кращу якість звуку, компактність і довговічність, і вже до середини 1990-х років вінілові платівки майже повністю зникли з масового ринку, залишившись популярними лише серед колекціонерів, діджеїв та аудіофілів.

Інструкція

Учаснику необхідно з використанням можливостей виключно табличного процесора створити засіб для конструювання дизайну вінілової платівки.

Вимоги до оформлення та роботи засобу:

- дизайн і загальна композиція мають відповідати наданим зразкам;
- користувач обирає рік зі списку в одному вікні, потім композицію – в іншому;
- також користувач може обрати бажаний колір етикетки платівки, використовуючи кнопки;
- інформація на етикетці платівки має оновлюватись відповідно до вибору користувача;
- у верхній частині етикетки при виборі має з'являтись ім'я виконавця / назва музичної групи;
- у нижній частині етикетки – назва музичного твору;
- вздовж нижнього краю етикетки – дата появи композиції на першому місці в європейських хіт-парадах;
- текст написів має розміщуватись у межах етикетки, не перекриваючи інших елементів.

У файлі *«Billboard-European Top.xlsx»* міститься інформація про найкращі музичні композиції та їх виконавців 1990–2000 років за версією хіт-параду журналу «Billboard».

Структура записів:

рік;

місяць та число;

назва музичного твору;

виконавець.

Результат виконання завдання зберегти у файл *«EuropeanTop.xlsx»*.

Примітка. Використання VBA та макросів заборонено.

Задача «Барабан»

Завдання виконується виключно засобами MS Excel
Результат роботи учасника потрібно зберегти у файлі «Барабан.xlsx».

Інколи важливо вчасно зробити перші кроки і спробувати себе в опануванні музичних інструментів. Можна скористатися іграшковими

музичними інструментами, однак сучасні технології дають змогу водночас застосувати інтерактивні тренажери. Звичайно, це непроста система, що поєднує програмний засіб і фізичну модель інструмента, але ми пропонуємо змоделювати простий алгоритм відслідковування правильності дій під час гри на музичному інструменті.

Учаснику пропонується з суворим дотриманням інструкції (файл «*Інструкція Excel Барабан.docx*») у табличному процесорі реалізувати роботу системи барабанного тренажера.

Ви хочете навчитися грати на барабанах з нуля? Чудово! Ймовірно, це тому, що вам властива надзвичайна ритмічність або ви не можете перестати клацати пальцями та відбивати ритм на поверхні меблів щоразу, коли чуєте ритмічні звуки. А можливо, ви просто розумієте, що барабанщики – найкрутіші учасники будь-якого музичного гурту.

Однак головне питання полягає зазвичай у тому, з чого почати. Яке обладнання знадобиться для початку? Чи потрібна одразу ціла ударна установка?

Спробуй насамперед створити барабанний тренажер, що складається з барабана та двох паличок, за допомогою яких можна відбивати ритм.

Інструкція

Учаснику пропонується виключно засобами електронних таблиць реалізувати роботу системи барабанного тренажера. Вигляд тренажера зображено на рис. 1.

Рис. 1

Керування паличками здійснюється за допомогою елементів керування (зліва – лівою паличкою, праворуч – правою), які можуть задавати верхнє та нижнє положення об'єкта (на рис. 2 відображено нижнє положення інструментів).

Рис. 2

Над інструментом розташовано текстовий елемент, який довільним чином задає користувачу, лівою чи правою паличкою потрібно виконати удар. Після кожного удару паличка має зайняти верхнє положення. Якщо завдання виконано правильно, на екрані з'являється схвальний рисунок у вигляді зеленої мітки, якщо ж користувач хибить – повідомлення про помилку (рис. 3).

Рис. 3

Файл із виконаною роботою збережіть із назвою *«Барабан.xlsx»*.

Задача «Українські композитори»

Завдання виконується виключно засобами MS PowerPoint.
Результат роботи учасника потрібно зберегти у файлі «Українські композитори.pptx».

Щоб інтерактивний рекламний постер ефективно виконував свою функцію, важливо привернути увагу до його змісту: певні статичні елементи на ньому мають бути акцентами, а використання гармонійної анімації має посилювати ефект сприйняття ідеї постера.

Учасникам пропонується створити інтерактивний рекламний постер з метою привернення уваги до творчості українських композиторів. Для максимальної ефективності певні статичні елементи на ньому мають бути акцентами, а використання гармонійної анімації має підсилювати ідею постера.

Інструкція

Учаснику необхідно на одному слайді суворо за зразком (файл *«Українські композитори.mp4»*) створити інтерактивну рекламну листівку про трьох видатних композиторів України.

Анімація має відбуватись автоматично при включенні режиму *«Показ слайдів»*.

Результат потрібно зберегти у файлі *«Українські композитори.pptx»*.

Список літератури

Інформатика

1. Жмурко О. І., Охріменко Т. О. Олімпіади з програмування. Прості задачі / Уманський держ. пед. ун-т імені Павла Тичини. Умань : Візаві, 2020. 298 с.

2. Караванова Т. П. Олімпіадна інформатика : навч. посіб. Чернівці : Чернівецький нац. ун-т ім. Ю. Федьковича, 2024. 232 с.

3. Тарасенко-Клятченко О. В., Клятченко Я. М., Михайлюк О. С. Збірник задач до практичних занять з програмування. Київ : КПІ ім. Ігоря Сікорського, 2021. 43 с. URL: https://scs.kpi.ua/storage/2022/01/ programuvannya_1_zadachi_do_praktychnyh.pdf (дата звернення: 23.07.2025).

4. Форкун Ю. В., Длугунович Н. А. Інформатика : навч. посіб. Львів : Новий Світ-2000, 2024. 464 с.

5. Хрол Н. П., Харченко В. М., Коваленко Л. Б. Збірник задач та розв'язків із програмування / за заг. ред. Ю. М. Літоша, О. Є. Баранової, О. М. Смірнової. Чернігів : ЧОІППО імені К. Д. Ушинського, 2016. Ч. 2. 71 с.

Інформаційні технології

1. Білак Ю. Ю., Лавер В. О., Андрашко Ю. В., Лях І. М. Інформатика та інформаційні технології : практикум. Ужгород : ПП «АУТДОР-ШАРК», 2015. Ч. 1. 96 с.

2. Інформаційні технології : навч. посіб. / О. І. Зачек та ін. ; за ред. О. І. Зачека. Львів : Львівський державний університет внутрішніх справ, 2022. 432 с. URL: https://dspace.lvduvs.edu.ua/handle/1234567890/6995 (дата звернення: 22.05.2025).

3. Кравченко І. В. Інформаційні технології : підручник для студ. спеціальності «Автоматизація та комп'ютерно-інтегровані технології». Київ : КПІ ім. Ігоря Сікорського, 2022. 447 с.

4. Логінова Н. І., Трофименко О. Г., Яценко М. А., Латковська Т. А. Інформаційні технології : навч.-метод. посіб. Одеса, 2024. 152 с. URL: https://doi.org/10.32837/11300.27258 (дата звернення: 22.05.2025).

5. Штучний інтелект. Нейромережева обробка інформації: архітектури, навчання, застосування : навч. посіб. У 2 ч. / О. Г. Руденко та ін. ; за заг. ред. С. П. Євсеєва. Харків : НТУ «ХПІ» — Львів : «Новий Світ-2000», 2025. Ч. 1. 426 с. URL: https://ns2000.com.ua/shtuchnyy-intelekt-navchalnyy-posibnyk-u-2-kh-ch-ch-1/ (дата звернення: 22.05.2025).

Відомості про авторів

ІНФОРМАТИКА

Горошко Юрій Васильович	завідувач кафедри інформатики та обчислювальної техніки природничо-математичного факультету Національного університету «Чернігівський колегіум» імені Т. Г. Шевченка, доктор педагогічних наук, професор
Денисов Костянтин Ігорович	студент факультету комп'ютерних наук та кібернетики Київського національного університету імені Тараса Шевченка

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Бондік Ірина Георгіївна	вчитель інформатики Дніпровського наукового ліцею інформаційних технологій Дніпровської міської ради
Громко Григорій Юрійович	вчитель інформатики Комунального закладу «Нечаївський ліцей ім. Ю. І. Яновського» Компаніївської селищної ради Кропивницького району Кіровоградської області
Журибеда Оксана Анатоліївна	учитель інформатики Білоцерківського ліцею-гімназії № 12 Білоцерківської міської ради Київської області

Кузічев Микола Миколайович	вчитель інформатики Дніпровської гімназії № 37 Дніпровської міської ради, вчитель-методист
Кушнір Віталій Анатолійович	вчитель Комунального закладу «Капустянська гімназія» Новоушицької селищної ради Хмельницької області
Ніколаєв Тарас Геннадійович	учитель інформатики ліцею № 100 «Лідер» Дніпровської міської ради
Паламарюк Максим Русланович	інженер даних ТзОВ Н-ІКС
Шаравара Ольга Валеріївна	учитель інформатики ліцею № 100 «Лідер» Дніпровської міської ради

Навчальне видання

Завдання IV етапу
Всеукраїнських учнівських олімпіад
з навчальних предметів

2024/2025 навчальний рік

ІНФОРМАТИКА

ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Практикум

Редагування: *З. В. Пономаренко*, *Т. І. Рябокінь*
Дизайн обкладинки *О. А. Чекановська*
Верстання *О. А. Жупанська*

У виданні з навчальною метою
використані матеріали, в тому числі ілюстративні,
що містяться у вільному доступі в мережі Інтернет

Формат 60×84 1/16. Папір офс. 80 г/м^2.
Друк цифровий. Ум. друк. арк. 5,12.
Наклад 300 пр.

Видавництво: Національний центр «Мала академія наук України»,
Кловський узвіз, буд. 8, м. Київ, 01021
Свідоцтво суб'єкта видавничої справи:
ДК № 6999 від 04.12.2019