Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра систем штуч ного інтелекту

Звіт

про виконання розрахунково-графічних робіт блоку № 7

з дисципліни: «Основи програмування»

до:

ВНС Розрахунково-графічних робіт № 1-4

Практичних Робіт до блоку № 7

Виконав(ла):

Студент групи ШІ-13

Недосіка Назарій Вадимович

Львів 2024

Завдання 1. Розробити лінійний алгоритм для розв'язання задачі.

$$P = \frac{1 + sin^2(x+1)}{2 + \left|x - \frac{2x^3}{1 + x^2y^3}\right|} + x^4; Q = cos^2\left(arctg\frac{1}{z}\right),$$

Де x = 0,25; y = 0,79; z = 0,81.

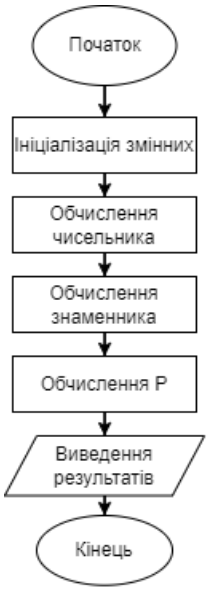
Розв'язок задачі

```cpp
#include <iostream>
#include <cmath>

using namespace std;

int main() {
    double x = 0.25;
    double y = 0.79;
    double z = 0.81;

    double numerator_P = 1 + pow(sin(x + 1), 2);
    double denominator_P = 2 + fabs(x - 2 * pow(x, 3) / (1 + pow(x, 2) * pow(y, 3)));
    double P = numerator_P / denominator_P + pow(x, 4);

    double Q = pow(cos(atan(1 / z)), 2);

    cout << "P = " << P << endl;
    cout << "Q = " << Q << endl;

    return 0;
}
```

Результат виконання

```
P = 0.860141
Q = 0.396172
```

Блок схема до завдання



Завдання 2. Розробити алгоритм, що розгалужується для розв'язання задачі номер якої відповідає порядковому номеру студента в журналі викладача.

$$y = \begin{cases} \cos(ax+2), & x < 2, \\ tg|x-2a|, & x \le 2; \end{cases} \text{ де } x \in [0{,}5; 3{,}1]; h_x = 0{,}2,$$

$a -$ має початкове значення 0,1 і змінюється одночасно зі змінною x з кроком $h_x = 0{,}3$.

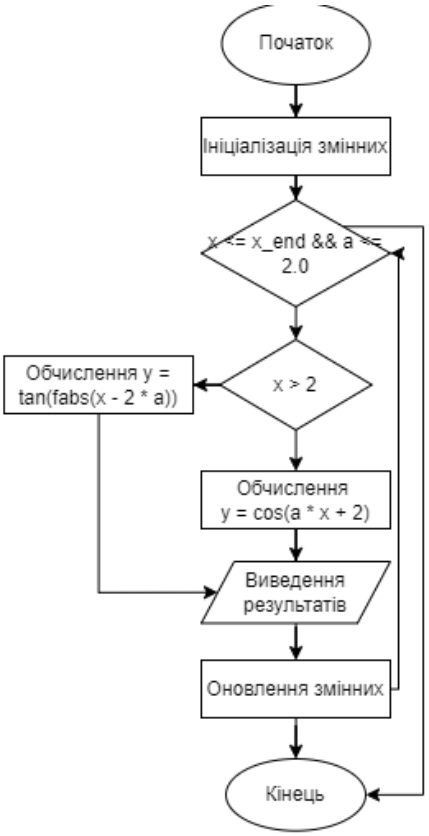
Розв'язок задачі

```cpp
#include <iostream>
#include <cmath>

using namespace std;

int main() {
    double x_start = 0.5, x_end = 3.1, hx = 0.2;
    double a_start = 0.1, ha = 0.3;
    double x = x_start, a = a_start;

    while (x <= x_end && a <= 2.0) {
        double y;

        if (x > 2) {
            y = cos(a * x + 2);
        } else {
            y = tan(fabs(x - 2 * a));
        }

        cout << "x = " << x << ", a = " << a << ", y = " << y << endl;

        x += hx;
        a += ha;
    }

    return 0;
}
```

Результат виконання

```
x = 0.5, a = 0.1, y = 0.309336
x = 0.7, a = 0.4, y = 0.100335
x = 0.9, a = 0.7, y = 0.546302
x = 1.1, a = 1, y = 1.26016
x = 1.3, a = 1.3, y = 3.6021
x = 1.5, a = 1.6, y = -7.6966
x = 1.7, a = 1.9, y = -1.70985
```

Блок схема до завдання



Завдання 3. Написати программу згідно свого варіанту.

Варіант 16. Обчислення величини доходу по внеску.

Процентна ставка (% річних) і час зберігання (днів) задаються під час роботи програми. Нижче приведений вид екрану під час виконання програми, що рекомендується (дані, введені користувачем, виділені напівжирним шрифтом).

Обчислення величини доходу по внеску.

Введіть початкові дані: Величина внеску (грн.) > **2500**

Термін внеску (днів) > **30**

Процентна ставка (річних в %) > **20**

Дохід: 41.10 грн.

Сума, після закінчення терміну внеску: 2541.10 грн.

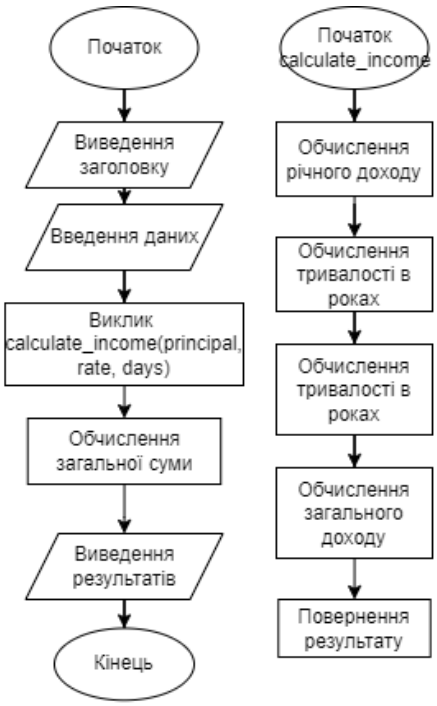
Розв’язок задачі

```cpp
#include <iostream>
#include <iomanip>

using namespace std;

double calculate_income(double principal, double rate, int days) {
    double yearly_income = principal * rate / 100;
    double years = days / 365.0;
    return yearly_income * years;
}

int main() {
    cout << "Обчислення величини доходу по внеску." << endl;

    double principal, rate;
    int days;

    cout << "Введіть суму внеску (грн): ";
    cin >> principal;
    cout << "Введіть процентну ставку (% річних): ";
    cin >> rate;
    cout << "Введіть час зберігання (днів): ";
    cin >> days;

    double income = calculate_income(principal, rate, days);
    double total = principal + income;

    cout << fixed << setprecision(2);
    cout << "\nДохід: " << income << " грн." << endl;
    cout << "Сума, після закінчення терміну внеску: " << total << " грн." << endl;

    return 0;
}
```

Результат виконання

```
Обчислення величини доходу по внеску.
Введіть суму внеску (грн): 2500
Введіть процентну ставку (% річних): 20
Введіть час зберігання (днів): 30

Дохід: 41.10 грн.
Сума, після закінчення терміну внеску: 2541.10 грн.
```

Блок схема до завдання



Завдання 4. Написати програму згідно свого варіанту.

Варіант 16. Скласти програму, яка генерує послідовності з 10 випадкових чисел в діапазоні від 1 до 10, виводить ці числа на екран і обчислює їх середнє арифметичне.

Розв’язок задачі

```cpp
#include <iostream>
#include <cstdlib>
#include <ctime>

using namespace std;

int main() {
    srand(static_cast<unsigned int>(time(0)));

    int sum = 0;
    const int count = 10;

    cout << "Згенеровані числа: ";

    for (int i = 0; i < count; ++i) {
        int random_number = rand() % 10 + 1;
        cout << random_number << " ";
        sum += random_number;
    }

    double average = static_cast<double>(sum) / count;

    cout << "\nСереднє арифметичне: " << average << endl;

    return 0;
}
```

Результат виконання

```
Згенеровані числа: 4 10 1 6 10 8 4 2 2 2
Середнє арифметичне: 4.9
```

Блок схема до завдання

Algotester task 1

Літня школа

Одного разу до Ужгорода на літню школу з алгоритмічного програмування приїхали nn студентів, що сформували kk команд. Відомо, що кожна команда складається з одного, двох або трьох студентів.

Вам необхідно визначити, скільки студентів було в кожній із команд.

Input

Єдиний рядок містить два цілих числа nn та kk — кількості студентів та команд.

Output

У єдиному рядку виведіть kk цілих чисел ajaj через пробіл. Тут ajaj — кількість студентів у jj-тій команді.

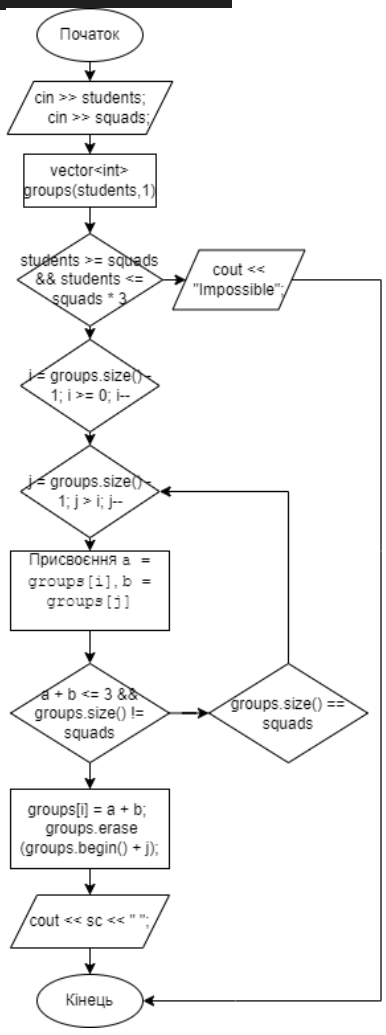
Якщо існує більше одного розв’язку — виведіть будь-який.

Якщо розв’язку не існує — виведіть **Impossible.**

Розв’язок задачі

```cpp
#include <iostream>
#include <vector>
using namespace std;

int main() {
    int students, squads;
    cin >> students;
    cin >> squads;

    vector<int> groups(students,1);

    if(students >= squads && students <= squads * 3){
        for(int i = groups.size() - 1; i >= 0; i--) {
            for(int j = groups.size() - 1; j > i; j--){

                int a = groups[i];
                int b = groups[j];

                if(a + b <= 3 && groups.size() != squads){
                    groups[i] = a + b;
                    groups.erase(groups.begin() + j);
                }
                else if(groups.size() == squads){
                    break;
                }
            }
        }
        for(int sc : groups){
            cout << sc << " ";
        }
    }
    else{
        cout << "Impossible";
    }
    return 0;
}
```

Блок схема до завдання



Результат виконання

```
7
4
1 1 2 3
```

Algotester task 2

Зуби

Мале Бісеня любить гострити зуби. А Зла Тітонька любить до нього підходити і питатися: «Що, зуби гостриш?». Бісеняті таке не дуже подобається, тому воно придумало робити таке.

У Малого Бісеняти є nn зубів. Кожен зуб має коефіцієнт загостреності aiai. Також існує межа загостреності kk. Якщо коефіцієнт загостреності певного зуба є більшим чи рівним межі загостреності, то такий зуб вважається загостреним.

Мале Бісеня хоче наступного разу, коли Зла Тітонька його щось запитає, показати їй якнайбільше загострених зубів, що розташовані поспіль.

Допоможіть Малому Бісеняті дізнатися, скільки найбільше зубів воно зможе показати.

Вхідні дані

У першому рядку задані два цілих числа nn та kk — кількість зубів та межа загостреності відповідно.

В другому рядку задано nn цілих чисел aiai — коефіцієнти загостреності зубів.

Вихідні дані

Єдине ціле число — відповідь на задачу.

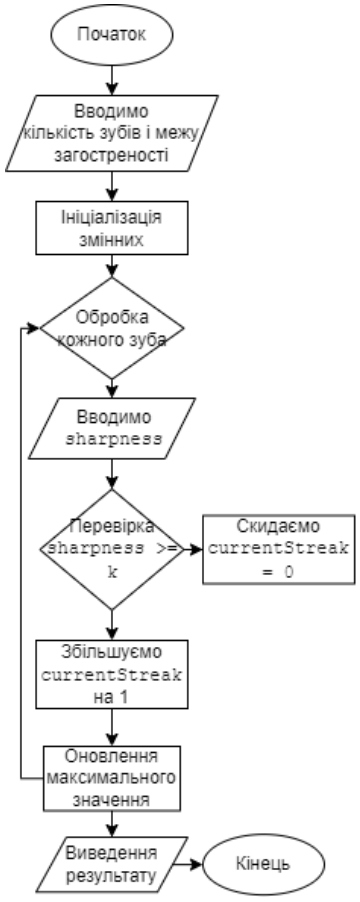
Розв'язок задачі

```cpp
#include <iostream>
#include <algorithm>
using namespace std;

int main() {
    int n, k;
    cin >> n >> k;

    int maxStreak = 0;
    int currentStreak = 0;

    for (int i = 0; i < n; ++i) {
        int sharpness;
        cin >> sharpness;

        if (sharpness >= k) {
            ++currentStreak;
        } else {
            currentStreak = 0;
        }

        maxStreak = max(maxStreak, currentStreak);
    }

    cout << maxStreak << endl;
    return 0;
}
```

Результат виконання

```
7 4
7 1 4 7 6 3 4
3
```

Блок схема до завдання



Algotester task 3

Лотерея

Одного разу двоє друзів, Віталік та Роман, вирішили зіграти в лотерею і навіть купили відповідний білет. На лотерейному білеті є прямокутна таблиця розміром n×mn×m. У кожній клітинці таблиці записане одне ціле число. Для участі в лотереї необхідно замалювати рівно одне число з таблиці та відіслати білет організаторам.

Віталік переконаний, що необхідно обрати найменше число, проте Роман абсолютно впевнений, що переможе найбільше. Білет у хлопців лише один, і вони довго не могли вирішити, як їм учинити. Після декількох днів активних суперечок та наукових дискусій на тему «Чому малі числа кращі, ніж великі» чи навпаки, друзі вирішили зробити так: спочатку Віталік обирає стовпець, а тоді Роман вибирає число з цього стовпця.

Ваше завдання визначити, яке число все-таки оберуть хлопці.

Вхідні дані

У першому рядку два цілі числа nn та mm — кількість рядків та стовпців лотерейної таблиці.

У наступних nn рядках по mm цілих чисел aijaij — jj-те число в ii-му рядку лотерейної таблиці.

Вихідні дані

У єдиному рядку виведіть число, яке виберуть Віталік та Роман.

Розв'язок задачі

```cpp
#include <iostream>
#include <vector>
#include <climits>
using namespace std;

int main() {
    int n, m;
    cin >> n >> m;

    vector<vector<int>> table(n, vector<int>(m));
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            cin >> table[i][j];
        }
    }

    int result = INT_MAX;
    for (int col = 0; col < m; ++col) {
        int columnMax = INT_MIN;
        for (int row = 0; row < n; ++row) {
            columnMax = max(columnMax, table[row][col]);
        }
        result = min(result, columnMax);
    }

    cout << result << endl;
    return 0;
}
```

Результат виконання

```
3 3
1 2 77
18 26 11
25 25 25
25
```

Блок схема до завдання

Algotester task 4

Марічка і печиво

Зібралися Зеник і Марічка разом з пластунами в похід. Похід — серйозна справа. Потрібно запастись продуктами харчування та розподілити їх споживання по днях так, щоб всім вистачило. Цього разу Зеник слідкує за тим, щоб печива вистачило аж до останнього дня походу. Зеник чітко знає, скільки пачок печива повинно залишитись кожного дня, і щовечора перераховує їх. Якщо Зеник побачить, що залишилось менше пачок, ніж повинно залишитись за його розрахунками, він неодмінно знайде того, хто з'їв забагато печива, і покарає його.

Марічка дуже любить печиво. Сьогодні, коли всі пластуни покинуть свої намети і підуть купатися в річку, Марічка планує непомітно з'їсти трохи печива. Звісно, Марічка не хоче бути покараною і дуже боїться, щоб Зеник не помітив пропажу.

Марічка підгледіла, скільки пачок печива є в рюкзаку Зеника. Також вона знає, скільки штук в кожній пачці. Марічці не терпиться дізнатися, скільки ж печива вона зможе з'їсти так, щоб Зеник не помітив. Зеник помітить пропажу печива з деякої пачки тоді і тільки тоді, коли Марічка повністю спустошить її.

Вхідні дані

У першому рядку задано одне натуральне число nn — кількість пачок печива.

У другому рядку задано nn натуральних чисел aiai — кількість штук печива в ii-й пачці.

Вихідні дані

У єдиному рядку виведіть одне ціле число — максимальну кількість штук печива, яку зможе з'їсти Марічка так, щоб Зеник не помітив цього.

Розв'язок задачі

```cpp
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    long long totalCookies = 0;
    int cookies;

    for (int i = 0; i < n; ++i) {
        cin >> cookies;
        if (cookies > 0) {
            totalCookies += cookies - 1;
        }
    }
    cout << totalCookies << endl;
    return 0;
}
```

Блок схема до завдання

Результат виконання

```
4
4 7 47 74
128
```

Контрольні запитання

1. Назвіть основні властивості алгоритму.
   1) Визначеність
   2) Дискретність
   3) Масовість
   4) Скінченність
   5) Вхідні дані
   6) Вихідні дані
   7) Ефективність

2. Що таке алгоритм?

   Алгоритм — це чіткий набір кроків або інструкцій, які потрібно виконати для досягнення певної мети або вирішення задачі.

3. Визначте основні етапи розробки алгоритмів.
   1) Аналіз задачі
   2) Проектування алгоритму
   3) Реалізація
   4) Тестування
   5) Оптимізація
4. Перелічить базові конструкції.
   1) Послідовність
   2) Розгалуження
   3) Цикл
5. Перелічить складні базові конструкції.
   1) Комбінація конструкцій
   2) Функції (процедури)

6. Дайте визначення конструкції розгалуження.
   Це частина алгоритму, яка дозволяє вибирати одну з двох або більше альтернатив залежно від виконання певної умови.

7. Дайте визначення конструкції цикл.
   Це частина алгоритму, яка дозволяє багаторазово виконувати один і той самий набір дій, поки виконуватиметься певна умова.

8. Сформулюйте правило виконання циклу з передумовою.
   Цикл з передумовоювиконується, якщо умова істинна на початку. Якщо умова не виконується одразу, цикл може не бути виконаний жодного разу.

9. Цикл з відомою кількістю повторів виконується задану кількість разів. Кількість повторів визначена на момент початку виконання циклу.

10. Що таке обчислювальна складність алгоритму?
    Це характеристика кількості ресурсів (часу, пам'яті), яку алгоритм використовує для виконання, залежно від величини вхідних даних. це характеристика кількості ресурсів (часу, пам'яті), яку алгоритм використовує для виконання, залежно від величини вхідних даних.

11. Як оцінити обчислювальну складність?
    За допомогою аналізу кількості операцій, які виконуються алгоритмом, та визначення залежності між розміром вхідних даних та кількістю цих операцій.

12. Рекурсивні функції. Переваги їх використання.
    Рекурсивна функція викликає саму себе для вирішення підзадач, що дозволяє розв'язувати складні проблеми шляхом поділу на менші частини.

    Переваги:
    1) Зручність при вирішенні задач з ітераціями або структурами даних, що мають рекурсивний характер (наприклад, дерева).
    2) Спрощення коду та зменшення необхідності використання складних циклів або стеків.
    3) Легкість у розв'язанні задач, де результат залежить від підзадач (наприклад, факторіал, числа Фібоначчі).

Висновок: Я закріпив практичні навички в розробці і дослідженні алгоритмів розв’язання задач.