# КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ТАРАСА ШЕВЧЕНКА



## ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

**Кафедра прикладних інформаційних систем**

**Звіт до лабораторної роботи №1**

# з курсу

**«Алгоритми і структури даних»**

*Студента 1 курсу*

*групи ПП-11 спеціальності 122 «Комп'ютерні науки» ОП «Прикладне програмування»*

%username%

*Викладач:*

д.е.н., к.т.н., проф. Плескач В.Л.

к.ф.-м.н., доц. Шолохов О.В.

## Київ – 202

1. **Назва роботи**

Лінійні алгоритми та алгоритми розгалуження

1. **Тема роботи**

Лінійні алгоритми та алгоритми розгалуження

1. **Мета роботи**

Дослідити та оволодіти навичками написання програм, що використовують лінійний та алгоритм розгалуження

4.**Теоритичні відомості**

1. Структурування алгоритмів - підхід, що дає можливість структурувати програму в цілому. Полягає цей підхід у представленні великої і складної задачі у вигляді декількох задач меншого обсягу, щоб врешті-решт модуль найвищого рівня (що відповідає за вирішення задачі в цілому) був достатньо простим. Фактично він забезбечує лише послідовність звертань до модулів, що вирішують задачі нижчого рівня, що в свою чергу теж звертаються до нижчих модулів, і так доки задача не буде досить легкою для виконання.
2. Структурування даних - інший підхід, що базується на інтегруванні базових типів та створенні більш складних, а також визначенні операцій над ними з метою звести їх до таких типів, що відповідають вхідним і вихідним даним задачі, а операції над ними реалізують в повному обсязі задачу проекту.
3. Інкапсуляція - технічний метод структуризації даних, що полягає в оформленні нового типу даних таким чином, що його внутрішня структура стає недоступною для програміста, який тепер може оперувати даними цього типу лише через виклик фукцій, визначених для цього типу. Приклад - класи (повністю закриті типи даних) та методи (функції, що працюють з їх внутрішньою структурою).
4. Концепція структур даних:

Під структурою даних загалом мають на увазі множину елементів даних та множину зв’язків між ними. У програмуванні та комп’ютерних науках структури даних є способами організації даних у комп’ютерах. Правильний підбір структур даних є надзвичайно важливим для ефективного функціонування відповідних алгоритмів їх обробки. Добре побудовані структури даних дозволяють оптимізувати використання машинного часу та пам'яті комп'ютера для виконання найкритичніших операцій.

1. Класифікація структур даних:

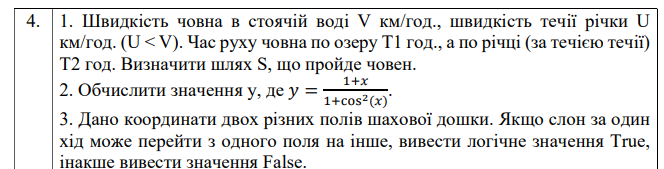
Структури даних діляться на прості (такі, що не можуть бути розділені на складові більше біта) та інтегровані (складовими яких є прості структури даних). Структура даних може бути зв’язною або незв’язною, а також мають ознаку змінності: зміна кількості елементів і/або зв’язків між елементами структури. За змінністю існують статичні, напівстатичні та динамічні структури.

1. Базові операції над структурами даних:

Існують чотири загальних операції для будь-якої структури даних: створення, знищення, вибір (доступ) та оновлення. Створення є базовим, але деколи охоплює також ініціалізацію структури. Знищення є його протилежністю, вибір дає програмістові доступ до власне даних структури, а оновлення дозволяє змінити значення даних у їх структурі.

5 .**Умова завдання**

(4 варіант)

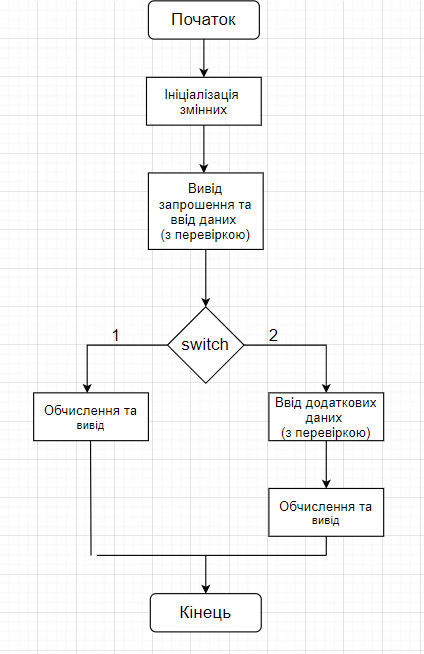


1. **Рішення**

Задача 1:

Ініціалізуємо змінні, виводимо запрошення та отримуємо дані від користувача. Перевіряємо, чи підходять введені дані; якщо все вірно, виводимо запрошення обрати один з варінтів розв’язку. Якщо обраний користувачем варіант не передбачає вводу додаткових даних, обчислюємо результат і виводимо на екран; якщо ж користувачеві потрібно ввести дані, то виводимо запрошення і перевіряємо ввід, після чого обчислюємо результат і виводимо на екран.

Блоксхема:



Текст програми:

static void firstTask() {

Console.WriteLine("Now executing the first task");

double boatSpeed = 0;

double flowSpeed = 0;

double time = 0;

double distance;

int switchFlag;

Console.WriteLine("Please enter the boat's speed:");

try {

boatSpeed = Convert.ToDouble(Console.ReadLine());

}

catch (Exception e) {

Console.WriteLine(e.Message);

}

Console.WriteLine("Please enter the time:");

try {

time = Convert.ToDouble(Console.ReadLine());

}

catch (Exception e) {

Console.WriteLine(e.Message);

}

Console.WriteLine("Please enter 1 if you want to calculate distance travelled by lake or 2 if you want to calculate distance travelled \nby river:");

switchFlag = Convert.ToInt16(Console.ReadLine());

switch (switchFlag) {

case 1:

distance = boatSpeed \* time;

Console.WriteLine("The distance is " + distance);

break;

case 2:

Console.WriteLine("Please enter the flow speed:");

try {

flowSpeed = Convert.ToDouble(Console.ReadLine());

}

catch (Exception e) {

Console.WriteLine(e.Message);

}

distance = (flowSpeed + boatSpeed) \* time;

Console.WriteLine("The distance is " + distance);

break;

default:

Console.WriteLine("d3f4lt");

break;

}

}

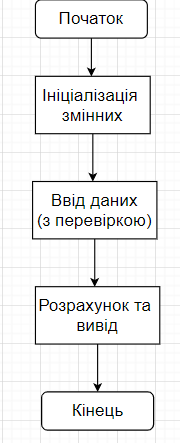
Результат роботи програми:

[screenshot]

Задача 2:

Ініціалізуємо змінні, в тому числі змінну cosOfX, присвоюючи їй значення методу Math.Cos(x), та змінну cosOfXSquared, що являє собою cosOfX, піднесену до квадрату за допомогою методу Math.Pow(). Виводимо запрошення, перевіряємо коректність введених користувачем даних, рахуємо та виводимо результат.

Блоксхема:



Текст програми:

static void secondTask() {

Console.WriteLine("Now executing the second task.");

double y;

double x = 0;

double cosOfX = Math.Cos(x);

double cosOfXSquared = Math.Pow(cosOfX, 2);

Console.WriteLine("Please enter the X:");

try {

x = Convert.ToDouble(Console.ReadLine());

} catch (Exception e) {

Console.WriteLine(e.Message);

}

y = (1 + x) / (1 + cosOfXSquared);

Console.WriteLine("Y equals " + y);

}

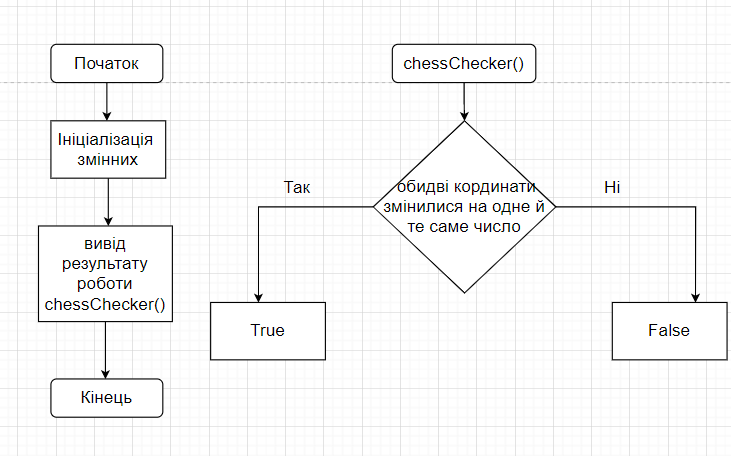
Результат роботи програми:

[screenshot]

Задача 3:

Ініціалізуємо змінні, виводимо запрошення на ввід даних. Введені дані перевіряємо та в залежності від отриманого результату повертаємо логічне значення Trure або False.

Блоксхема:



Текст програми:

static void thirdTask() {

Console.WriteLine("Now executing third task");

int x = 0;

int y = 0;

int x2 = 0;

int y2 = 0;

Console.WriteLine("Please enter X and Y of the first, then the second point:");

try {

x = Convert.ToInt32(Console.ReadLine());

y = Convert.ToInt32(Console.ReadLine());

x2 = Convert.ToInt32(Console.ReadLine());

y2 = Convert.ToInt32(Console.ReadLine());

} catch (Exception e) {

Console.WriteLine(e.Message);

}

bool chessChecker() {

int xResult = x2 - x;

int yResult = y2 - y;

if (xResult == yResult) {

return true;

} else {

return false;

}

}

Console.WriteLine(chessChecker());

}

Результат роботи програми:

[screenshot]

[screenshot]

Ці три методи викликаються у функції main, в якій згідно з завданням реалізоване меню:

static void Main(string[] args) {

int switchFlag = 0;

Console.WriteLine("Please enter 1, 2 or 3 to start executing tasks in corresponding order:");

try {

switchFlag = Convert.ToInt32(Console.ReadLine());

} catch (Exception e) {

Console.WriteLine(e.Message);

}

switch (switchFlag) {

case 1:

firstTask();

break;

case 2:

secondTask();

break;

case 3:

thirdTask();

break;

default:

Console.WriteLine("No such option.");

break;

}

}

1. **Висновки**

В результаті виконання даної лабораторної роботи я дослідив та оволодів навичками написання програм, що використовують лінійний та алгоритм розгалуження, навчився використовувати останні алгоритми для виконання поставлених задач. Вважаю дану лабораторну роботу виконаною в повному обсязі.