МІНІСТЕРСТВО ОСВІТИ і НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп'ютерних систем

Лабораторна робота №3

з дисципліни «**Паралельне та розподілене обчислення»**

Тема: «Емуляція системи команд в обчислювальній системі з мікропрограмним керуванням»

Виконав:

студент групи

КВ84

Воєводін І.П.

Перевірив:

Київ – 2021

Постановка завдання для програми мовою C

1. Написати програму розв’язання задачі пошуку (за варіантом) у двовимірному масиві (матриці) одним з алгоритмів методу лінійного пошуку.

2. Розміри матриці m та n взяти самостійно у межах від 7 до 10. Розмір матриці повинен задаватися аргументом запуску програми.

3. Програма обов'язково повинна бути написана і структурована наступним чином:

a) оголошення структур даних (typedef) повинно бути зроблено у окремому заголовочному файлі;

b) повинно бути щонайменше три файли із вихідним кодом (не враховуючи необхідні заголовочні файли), що міститимуть реалізації функцій введення (випадкові значення, наперед сортовані значення, з клавіатури), обробки, та виведення на друк (pretty_print) елементів матриці;

c) для виконання завдання обробки елементів матриці повинно бути написано дві різні функції:

1) з додатковими операторами виведення налагоджувальної інформації на друк (debug-версія);

2) з виконанням заданих дій без додаткового виведення налагоджувальної інформації (release-версія).

Вибір функції повинен робити користувач при запуску програми через аргумент запуску. Наприклад, опція -d вмикає debug. 1. Для компіляції написаної багатофайлової програми написати окремий make-файл, причому:

a) при зміні одного із вихідних файлів повинен перекомпільовуватися лише цей файл (а також відбуватися дії, необхідні для генерації бінарного файлу);

b) при видаленні бінарного файлу та незмінних вихідних файлах повинна відбуватися лише лінковка;

c) забезпечити окрему ціль для очистки згенерованих файлів; 12/18 1.

Вміти компілювати написану багатофайлову програму двома способами:

a) за допомогою однієї команди gcc;

b) за допомогою make-файлу. 1.

Виконати тестування та налагодження програми на комп’ютері. При тестуванні програми необхідно підбирати такі вхідні набори початкових значень матриці, щоб можна було легко відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Постановка завдання для програми мовою Java

1. Написати консольну програму розв’язання задачі пошуку (за

варіантом) у двовимірному масиві (матриці) одним з алгоритмів методу лінійного пошуку.

2. Розміри матриці m та n взяти самостійно у межах від 7 до 10.

3. При написанні програми повинно бути щонайменше три класи, один із яких буде відповідати за пошук елементу в матриці, другий відповідати за ввід-вивід матриці, а третій — головний клас, що міститиме метод main.

4. Для компіляції та запуску написаної програми написати окремий make-файл, причому забезпечити окремі цілі для очистки згенерованих файлів, а також генерації JAR-архіву.

5. Вміти компілювати написану програму двома способами:

a) за допомогою однієї команди javac;

b) за допомогою make-файлу. 1.

Виконати тестування та налагодження програми на комп'ютері. При тестуванні програми необхідно підбирати такі вхідні набори початкових значень матриці, щоб можна було легко відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Варіант 29

Задано матрицю дійсних чисел A[n,n]. У побічній діагоналі матриці визначити присутність заданого дійсного числа X> - це команда яку необхідно виконати. Команди виконуються і його місцезнаходження (координати).

Програма мовою С

Input. h

```
#ifndef _INPUT_H
#include <stdio.h>
#include <stdlib.h>

#define N 10

void HandInput(const int n, int** A);
void SortedInput(const int n, int** A);
void RandInput(const int n, int** A);

#endif // !_INPUT_H
```

Input.c

```
#include "Input.h"
```

```
void HandInput(const int n, int** A)
{
    int val;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n ; j++)
        {
            printf("Input value for [%d][%d]:", i, j);
            scanf("%d", &val);
            (*(*A + i * n + j)) = val;
        }
    }
}

void SortedInput(const int n, int** A)
{
    int matrix[N][N] = {
        { 21, 34, 56, 22, 76, 90, 36, 47, 83, 14},
        { 45, 10, 17, 84, 62, 49, 58, 55, 13, 98},
        { 16, 26, 82, 67, 15, 65, 12, 24 , 15, 33},
        { 79, 13, 57, 44, 38, 94, 11, 47, 49, 86},
        { 46, 12, 64, 83, 47, 23, 10, 21, 30 ,66},
        { 34, 75, 14, 73, 52, 39, 27, 32, 11, 78},
```

```
        { 45, 27, 48, 18, 95, 60, 71, 76, 88 ,37},
          { 76, 10, 46, 83, 27, 18, 72, 92, 37, 58},
          { 14, 41, 62, 72, 17, 48, 39, 94, 93, 77},
          { 15, 67, 73, 45, 47, 37, 74, 59, 37, 28}
    };
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            (*(*A + i * n + j)) = matrix[i][j];
        }
    }
}

void RandInput(const int n, int** A)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            (*(*A + i*n + j)) = rand() % 100;
        }
```

```
    }

}
```

Print.h

```
        #ifndef _PRINT_H
#include "Input.h"

void pretty_print(const int n, int* A);
void menu(const int n, int** A);

#endif // !_PRINT_H
```

Print.c

```
#include "Print.h"

void pretty_print(const int n, int* A)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            printf("%d\t", *(A + i*n + j));
        }
        printf("\n\n");
    }
}

void menu(const int n, int** A)
{
    int var;
    printf("Choose type of matrix filler:\n 1)Keyboard\n 2)Rand\n 3)Sorted\n Print
number to choose:");
    scanf("%d", &var);
    switch (var)
    {
    case 1: {HandInput(n,A); pretty_print(n, *A); break; }
    case 2: {RandInput(n, A); pretty_print(n, *A); break; }
    case 3: {SortedInput(n, A); pretty_print(n, *A); break; }
        default:
        { printf("Wrong input");
        break; }
    }

}
```

Search.h

```
#ifndef _SEARCH_H
#include "Input.h"

void ReleaseSearch(int x, int* row_x, int* clmn_x, const int n, int* A);
void DebugSearch(int x, int* row_x, int* clmn_x, const int n, int* A);

#endif // !_SEARCH_H
```

Search.c

```
#include "Search.h"

void ReleaseSearch(int x, int* row_x, int* clmn_x, const int n, int* A)
{
    for (int i = 0; i < n; i++)
    {
        int j = n - i - 1;

        if (*(A + i * n + j) == x)
        {
            *row_x = i + 1;
            *clmn_x = j + 1;
            return;
        }
        else
        {
            if ((i == n - 1) && (j == 0))
            {
                *row_x = 0;
                *clmn_x = 0;
            }
        }
    }
}

void DebugSearch(int x, int* row_x, int* clmn_x, const int n, int* A)
{
    for (int i = 0; i < n; i++)
    {
        int j = n - i - 1;

        printf("Searching...\n Current value: %d\n", *(A + i * n + j));
        if (*(A + i * n + j) == x)
        {
            *row_x = i + 1;
```

```
            *clmn_x = j + 1;
            printf("Value %d was found at [%d][%d]", x, *row_x, *clmn_x);
            return;
        }
        else
        {
            if ((i == n - 1) && (j == 0))
            {
                printf("Value %d was not found!", x);
                *row_x = 0;
                *clmn_x = 0;
            }
        }
    }
}
```

Main.c

```
#include "Print.h"
#include "Search.h"
#include <getopt.h>

int main(int argc, char** argv)
{
        const int n = atoi(argv[1]);
        if(n == 0)
        {
                printf("Input matrix size!");
                return 0;
        }
        int* A = (int*)malloc(n * n * sizeof(int));

        int row, clmn,x;
        menu(n, &A);

        printf("Input value to search:");
        scanf("%d",&x);
        printf("\n");

        int c = 0;
        const char* short_options = "dr";
        static struct option long_options[] = {{"debug", no_argument, 0, 'd'},
{"release", no_argument, 0, 'r'}, {0, 0, 0, 0}};
        int option_index = 0;
        while((c = getopt_long(argc, argv, short_options,
long_options,&option_index)) != -1)
```

```
        {
                switch (c)
                {
                case 'd':
                        DebugSearch(x, &row, &clmn, n, A);
                        break;
                case 'r':
                        ReleaseSearch(x, &row,&clmn, n, A);
                printf("Row:%d, Clmn:%d", row, clmn);
                        break;
                default:
                        printf("Wrong option input!");
                        break;
                }
        }

        free(A);
        return 0;
}
```

makefile

```
.PHONY: greet build rebuild run clean

%.o:%.c

greet:
        @echo "Terminating make - please specify target explicitly"
        @echo "    build   : fast rebuild / build"
        @echo "    rebuild : full rebuild"
        @echo "    run     : run after fast rebuild / build"
        @echo "    clean   : perform full clean"

build:  prog

rebuild: clean prog

run:
        ./prog $(N) $(OPT)
clean:
        rm -rvf *.o prog

prog: main.o Print.o Search.o Input.o
        gcc -o prog main.o Print.o Search.o Input.o

Input.o: Input.c Input.h
```

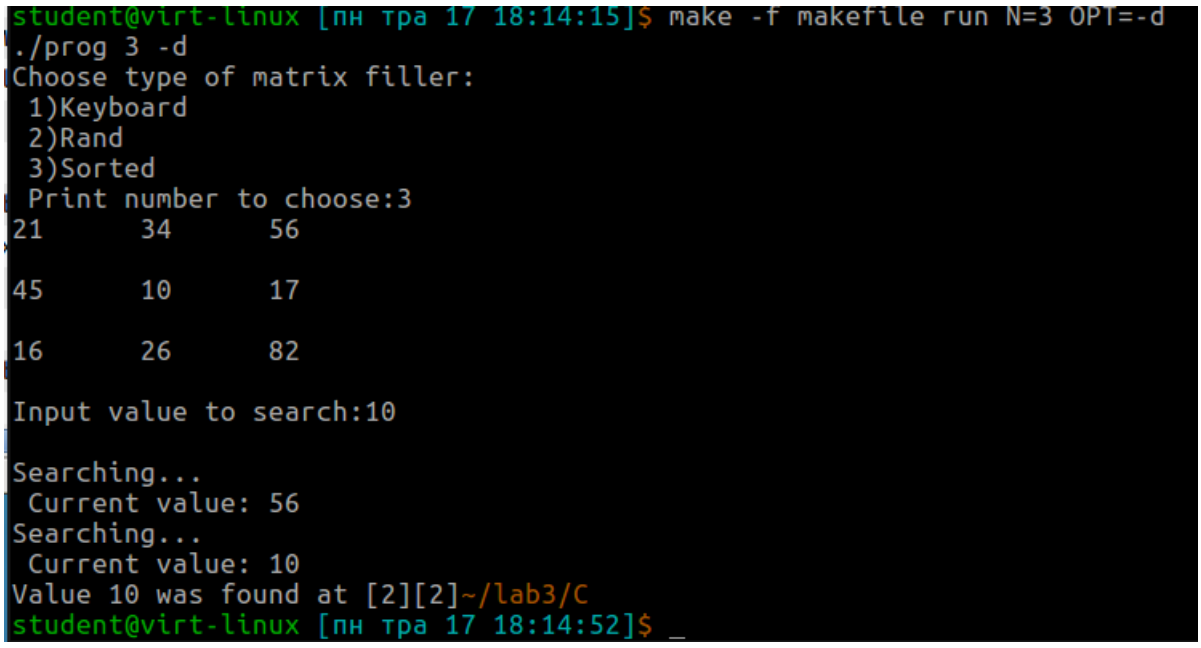
```
	gcc -c -o Input.o Input.c

Print.o: Print.c Print.h Input.h
	gcc -c -o Print.o Print.c

Search.o: Search.c Search.h Input.h
	gcc -c -o Search.o Search.c

main.o: main.c Print.h Search.h Input.h
	gcc -c -o main.o main.c
```

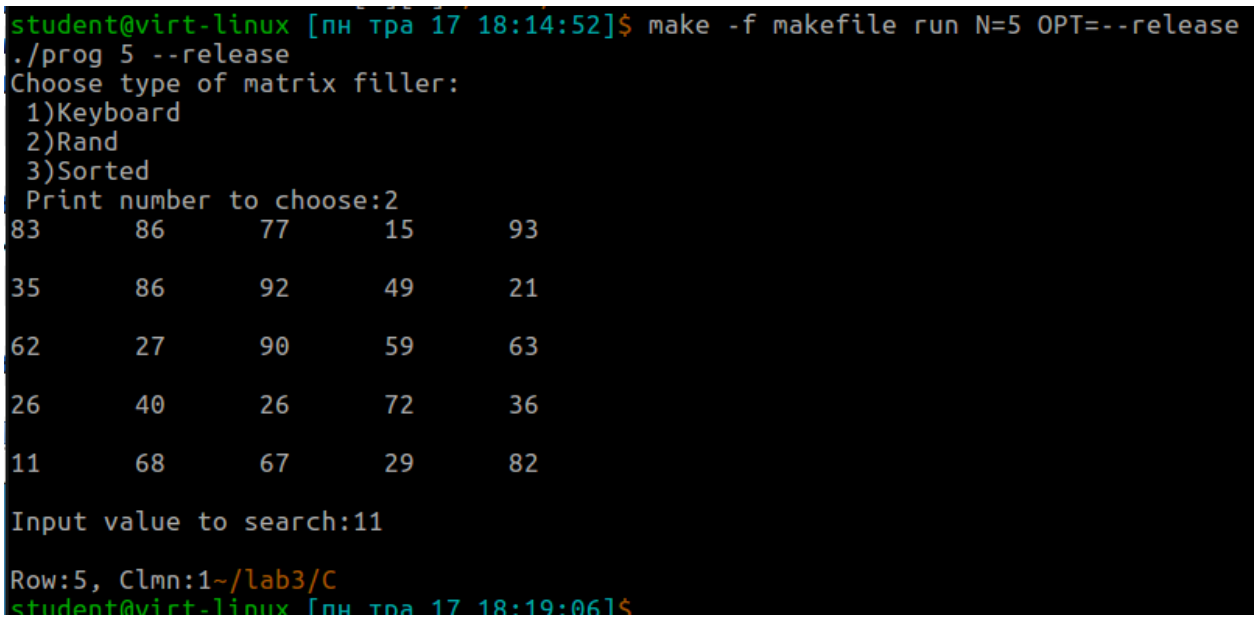
Компіляція

```
gcc -o main main.c Input.c Print.c Search.c
```

Запуск

./main ‘#’ ‘opt’





Програма мовою Java

Print_input.java

```
package lab3.java.Print_InputClass;
import java.util.Scanner;

public class print_input {
   public static void HandInput(final int n, int[][] A)
   {
      Scanner in = new Scanner(System.in);
      int val;
      for (int i = 0; i < n; i++)
      {
         for (int j = 0; j < n ; j++)
         {
            System.out.printf("Input value for [%d][%d]:", i, j);
            val = in.nextInt();
            A[i][j] = val;
         }
      }
      in.close();
   }

   public static void SortedInput(final int n, int[][] A)
   {
      int[][] matrix= {
      { 21, 34, 56, 22, 76, 90, 36, 47, 83, 14},
      { 45, 10, 17, 84, 62, 49, 58, 55, 13, 98},
      { 16, 26, 82, 67, 15, 65, 12, 24 , 15, 33},
      { 79, 13, 57, 44, 38, 94, 11, 47, 49, 86},
```

```
        { 46, 12, 64, 83, 47, 23, 10, 21, 30 ,66},
        { 34, 75, 14, 73, 52, 39, 27, 32, 11, 78},
        { 45, 27, 48, 18, 95, 60, 71, 76, 88 ,37},
        { 76, 10, 46, 83, 27, 18, 72, 92, 37, 58},
        { 14, 41, 62, 72, 17, 48, 39, 94, 93, 77},
        { 15, 67, 73, 45, 47, 37, 74, 59, 37, 28}
    };
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                A[i][j] = matrix[i][j];
            }
        }
    }

    public static void RandInput(int n, int[][] A)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                A[i][j] = (int)(Math.random() * 100);
            }
        }
    }

    public static void pretty_print(final int n, int[][] A)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                System.out.printf("%d\t", A[i][j]);
            }
            System.out.print("\n\n");
        }
    }
    public static void menu(final int n, int[][] A)
    {
        Scanner in = new Scanner(System.in);
        int var;
        System.out.print("Choose type of matrix filler:\n 1)Keyboard\n 2)Rand\n
3)Sorted\n Print number to choose:");
        var = in.nextInt();
```

```
        switch (var) {
            case 1:
                HandInput(n, A);
                pretty_print(n, A);
                break;
            case 2:
                RandInput(n, A);
                pretty_print(n, A);
                break;
            case 3:
                SortedInput(n, A);
                pretty_print(n, A);
                break;
            default:
                System.out.print("Wrong input");
                break;
        }
    }
}
```

Search.java

```
package lab3.java.SearchClass;

public class search {
    public static void ReleaseSearch(int x, int row_x, int clmn_x, final int n, int[][]
A)
    {
        for (int i = 0; i < n; i++)
        {
            int j = n - i - 1;
            if (A[i][j] == x)
            {
                row_x = i + 1;
                clmn_x = j + 1;
                    System.out.printf("Row: %d, Clmn: %d",row_x, clmn_x);
                return;
            }
        else
            {
                if ((i == n - 1) && (j == 0))
                {
                    System.out.println("Value not found");
                }
            }
        }
```

```
    }

    public static void DebugSearch(int x, int row_x, int clmn_x, final int n, int[][] A)
    {
        for (int i = 0; i < n; i++)
        {
            int j = n - i - 1;

            System.out.printf("Searching...\n Current value: %d\n", A[i][j]);
            if (A[i][j] == x)
            {
                row_x = i + 1;
                clmn_x = j + 1;
                System.out.printf("Value %d was found at [%d][%d]", x, row_x,
clmn_x);
                return;
            }
        else
            {
                if ((i == n - 1) && (j == 0))
                {
                    System.out.printf("Value %d was not found!", x);
                }
            }
        }
    }
}
```

main.java

```
package lab3.java.MainClass;
import java.util.Scanner;
import lab3.java.SearchClass.search;
import lab3.java.Print_InputClass.print_input;

public class main {

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = Integer.parseInt(args[0]);
        if(n == 0)
        {
                System.out.println("Input matrix size!");
                return;
         }
         int[][] A = new int[n][n];
```

```
        int row = 0, clmn = 0,x;
        print_input.menu(n,A);

        System.out.print("Input value to search:");
        x = in.nextInt();
        System.out.print("\n");

          String opt = args[1];
             switch (opt) {
                case "-d":
                   search.DebugSearch(x, row, clmn, n, A);
                   break;
                case "-r":
                   search.ReleaseSearch(x, row, clmn, n, A);
                   break;
                default:
                   System.out.print("Wrong option input!");
                   break;
             }
      }
}
```

manifest

```
.PHONY: greet build run jar run-jar clean

greet:
        @echo "Terminating make - please specify target explicitly"
        @echo "    build   : compile java program"
        @echo "    run     : run java program using class tree"
        @echo "    jar     : pack java program into an archive for easy distribution"
        @echo "    run-jar : run java program using jar archive"
        @echo "    clean   : perform full clean"

build:
        javac lab3/java/MainClass/main.java

run: build
        java lab3.java.MainClass.main $(N) $(OPT)

jar: build
        jar cvfm main.jar manifest `find . -name "*.class"`

run-jar: jar
        java -jar main.jar $(N) $(OPT)
```

```
clean:
	rm -vf *.jar
	find . -name "*.class" -delete -printf "removed '%p'\n"
```

Компіляція

javac lab3/java/MainClass/main.java

Запуск

java lab3.java.MainClass.main ‘#’ ‘option’

```
student@virt-linux [пн тра 17 18:21:25]$ make build
javac lab3/java/MainClass/main.java
~
student@virt-linux [пн тра 17 18:21:59]$ make run N=7 OPT=-d
javac lab3/java/MainClass/main.java
java lab3.java.MainClass.main 7 -d
Choose type of matrix filler:
 1)Keyboard
 2)Rand
 3)Sorted
 Print number to choose:2
82      7       92      20      96      19      12

84      7       10      27      43      5       44

34      70      7       53      53      86      98

28      58      32      90      97      66      55

24      84      64      14      84      51      45

94      46      34      56      49      5       91

64      88      8       40      52      50      43

Input value to search:12

Searching...
 Current value: 12
Value 12 was found at [1][7]~
```

```
student@virt-linux [пн тра 17 19:01:23]$ make run N=2 OPT=-r
javac lab3/java/MainClass/main.java
java lab3.java.MainClass.main 2 -r
Choose type of matrix filler:
 1)Keyboard
 2)Rand
 3)Sorted
 Print number to choose:1
Input value for [0][0]:1
Input value for [0][1]:2
Input value for [1][0]:3
Input value for [1][1]:4
1       2

3       4

Input value to search:2

Row: 1, Clmn: 2~
```