

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих комп’ютерних систем**

**Лабораторна робота №1**

з дисципліни

**«Паралельні та розподілені обчислення»**

**ТЕМА: «Робота з компіляторами мов С та Java в режимі командного рядка»**

Виконав: студент ІІ курсу

ФПМ групи КВ-04

Устименко Ілля Віталійович

Перевірила:

Київ – 2022

**Постановка задачі**

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

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

Програма обов'язково повинна бути написана і структурована таким 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).

4. Вибір функції повинен робити користувач при запуску програми через аргумент запуску. Наприклад, опція -d вмикає debug режим.

5. 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 повинне відбуватися лише лінкування (компоновка бінарного файлу з об’єктних);

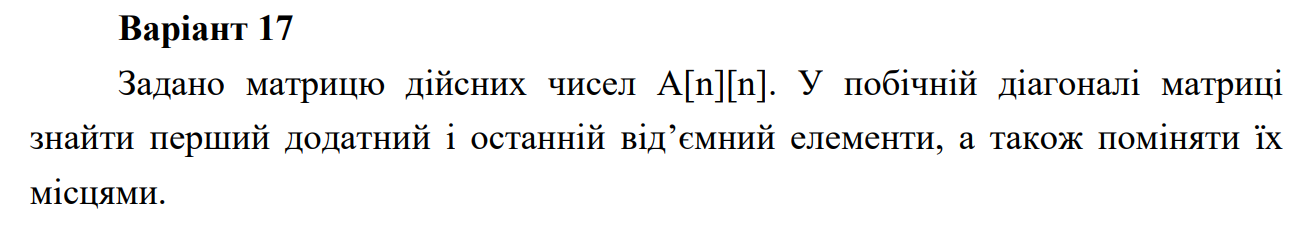
c) забезпечити окрему ціль для очистки згенерованих файлів;

6. Забезпечити можливість компіляції написаної багатофайлової програми двома способами:

a) за допомогою однієї команди gcc;

b) за допомогою make-файлу. Марченко О.І., Марченко О.О.

7.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.



Програма мовою C

main.c:

#include "process.h"

#include "action.h"

#include "work\_with\_memory.h"

**int** main(**int** argc,**char**\*argv[])

{

**if**(work\_with\_arg(argc,argv) == 'd')

{

make\_arr();

debug();

}

**else**

{

make\_arr();

realese();

}

del\_arr();

}

action.h:

#ifndef ACTION\_H

#define ACTION\_H

**void** realese();

**void** debug();

#endif

action.c:

#include"input.h"

#include"pretty\_print.h"

#include"process.h"

#include <stdio.h>

**void** realese()

{

printf("realese\n");

printf("-----With random array and relese function--------\n");

inputRand();

outputArr();

findAndSwapRelease();

outputArr();

printf("------With sorted array and release function------------\n");

inputSorted();

outputArr();

findAndSwapRelease();

outputArr();

printf("--------With array from keyboard and release function----------\n");

inputKeyboard();

findAndSwapRelease();

outputArr();

}

**void** debug()

{

printf("debug\n");

printf("-----With random array and debug function--------\n");

inputRand();

outputArr();

findAndSwapDebug();

outputArr();

printf("------With sorted array and release function------------\n");

inputSorted();

outputArr();

findAndSwapDebug();

outputArr();

printf("--------With array from keyboard and release function----------\n");

inputKeyboard();

findAndSwapDebug();

outputArr();

}

input.h:

#ifndef INPUT\_H

#define INPUT\_H

**void** inputRand(**void**);

**void** inputSorted(**void**);

**void** inputKeyboard(**void**);

#endif

input.c:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include "typedef.h"

**void** inputRand()

{

srand(time(0));

**for**(**int** i = 0; i < how\_n; ++i)

{

**for**(**int** j = 0; j < how\_n; ++j)

{

arr[i][j]=-40 + rand() % 100;

}

}

}

**void** inputSorted()

{

**int** b = 0;

**for**(**int** i = 0; i < how\_n ; ++i)

{

**for**(**int** j = 0; j < how\_n; ++j)

{

arr[i][j] = b;

++b;

}

}

}

**void** inputKeyboard()

{

**for**(**int** i = 0; i < how\_n; ++i)

{

**for**(**int** j = 0; j < how\_n; ++j)

{

scanf("%d",&arr[i][j]);

}

}

}

process.h:

#ifndef PROCESS\_H

#define PROCESS\_H

**void** findAndSwapRelease(**void**);

**void** findAndSwapDebug(**void**);

**char** work\_with\_arg(**int** argc,**char**\*argv[]);

#endif

process.c:

#include <stdlib.h>

#include "typedef.h"

#include <getopt.h>

#include <stdio.h>

**char** work\_with\_arg(**int** argc,**char**\*argv[]){

**char** deb\_or\_rea='r';

**char** take\_char;

**while**((take\_char = getopt(argc,argv,"drn:"))!=-1)

{

**switch**(take\_char)

{

**case**'r':

deb\_or\_rea = 'r';

**break**;

**case**'d':

deb\_or\_rea = 'd';

**break**;

**case**'n':

how\_n = atoi(optarg);

**break**;

}

}

**return** deb\_or\_rea;

}

**void** findAndSwapRelease(){

**int** tmp,firstPositive,ifpos = -1,lastNegative,ilneg = -1;

**for**(**int** i = 0; i < how\_n ; ++i)

{

**if**(arr[how\_n-1-i][i] >= 0)

{

firstPositive = arr[how\_n-1-i][i];

ifpos = i;

**break**;

}

}

**for**(**int** i = 0; i < how\_n; ++i) {

**if**(arr[i][how\_n-1-i] < 0)

{

lastNegative = arr[i][how\_n-1-i];

ilneg = i;

**break**;

}

}

**if**((ifpos != -1)&&(ilneg != -1)){

tmp = arr[how\_n-1-ifpos][ifpos];

arr[how\_n-1-ifpos][ifpos] = arr[ilneg][how\_n-1-ilneg];

arr[ilneg][how\_n-1-ilneg] = tmp;

printf("\nswap\n");

}**else** **if**(ifpos == -1){printf("\nno\_Positive\_in\_a\_mass\n\n");}**else**{printf("\nno\_Negative\_in\_a\_mass\n\n");}

}

**void** findAndSwapDebug(){

**int** tmp,firstPositive,ifpos = -1,lastNegative,ilneg = -1;

**for**(**int** i = 0; i < how\_n ; ++i)

{

printf("arr[%d][%d]=%d\n",how\_n-1-i,i,arr[how\_n-1-i][i]);

**if**(arr[how\_n-1-i][i] >= 0)

{

firstPositive = arr[how\_n-1-i][i];

ifpos = i;

printf("firstPositive = %d\t",firstPositive);

printf("ifpos=%d",ifpos);

**break**;

}

}

printf("\n\n");

**for**(**int** i = 0; i < how\_n; ++i) {

printf("arr[%d][%d]=%d\n",i,how\_n-1-i,arr[i][how\_n-1-i]);

**if**(arr[i][how\_n-1-i] < 0)

{

lastNegative = arr[i][how\_n-1-i];

ilneg = i;

printf("lastNegative =%d\t",lastNegative);

printf("ilneg=%d\n\n",ilneg);

**break**;

}

}

**if**((ifpos != -1)&&(ilneg != -1)) {

tmp = arr[how\_n-1-ifpos][ifpos];

arr[how\_n-1-ifpos][ifpos] = arr[ilneg][how\_n-1-ilneg];

arr[ilneg][how\_n-1-ilneg] = tmp;

printf("sideways diagonal after swapping:");

**for**(**int** i = 0; i < how\_n; i++){

printf("%d\t",arr[how\_n-1-i][i]);

}

}**else** **if**(ifpos == -1){printf("\nno\_Positive\_in\_a\_mass\n\n");}**else**{printf("\nno\_Negative\_in\_a\_mass\n\n");}

printf("\n\n");

}

pretty\_print.h:

#ifndef PRETTY\_PRINT\_H

#define PRETTY\_PRINT\_H

**void** outputArr();

#endif

pretty\_print.c:

#include <stdio.h>

#include "typedef.h"

**void** outputArr()

{

**for**(**int** i = 0; i < how\_n; ++i)

{

**for**(**int** j = 0; j < how\_n ; ++j)

{

printf("%d\t",arr[i][j]);

}

printf("\n");

}

printf("\n");

}

typedef.h:

#ifndef TYPEDEF\_H

#define TYPEDEF\_H

**extern** **int** how\_n;

**extern** **int**\*\* arr;

#endif

typedef.c:

#include "typedef.h"

**int**\*\* arr;

**int** how\_n = 10;

work\_with\_memory.h:

#ifndef WORK\_WITH\_MEMORY\_H

#define WORK\_WITH\_MEMORY\_H

**void** make\_arr();

**void** del\_arr();

#endif

work\_with\_memory.c:

#include<stdlib.h>

#include"typedef.h"

**void** make\_arr()

{

arr = (**int**\*\*)malloc(how\_n \* **sizeof**(**int**\*));

**for**(**int** i = 0; i < how\_n; ++i)

{

\*(arr+i) = (**int**\*)malloc(how\_n \* **sizeof**(**int**));

}

}

**void** del\_arr()

{

**for**(**int** i = 0;i < how\_n; ++i)

{

free(arr[i]);

}

free(arr);

}

Командні рядки для компілювання та запуска програми мовою С

термінал:

gcc -o create main.c action.c input.c pretty\_print.c process.c typedef.c work\_with\_memory.c

./create

make build-скомпілювати програму,або якщо якихось файли є-скомпілювати,ті яких нема

make rebuild-перекомпілювати повністю програму

make run-скомпілювати і запустити

make debug-скомпілювати в режимі дебаг

make realese-скомпілювати в режимі-реліз

make clean-очистити об’єктні файли і сам бінарний файл

визов дебаг режиму чи розміру матриці

./create -r -n4-приклад

./create -(вид режиму r чи d) -n(розмырнысть матриці число)

makefile:

.PHONY:build clean rebuild run greet debug realese

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: create

rebuild:clean create

run:build

./create

debug: build

./create -d

realese: build

./create -r

clean:

rm -rvf \*.o create

main.o:main.c process.h action.h work\_with\_memory.h

gcc -c -o main.o main.c

process.o:process.c **typedef**.h

gcc -c -o process.o process.c

action.o: action.c input.h process.h pretty\_print.h

gcc -c -o action.o action.c

work\_with\_memory.o: work\_with\_memory.c **typedef**.h

gcc -c -o work\_with\_memory.o work\_with\_memory.c

input.o:input.c **typedef**.h

gcc -c -o input.o input.c

pretty\_print.o:pretty\_print.c **typedef**.h

gcc -c -o pretty\_print.o pretty\_print.c

**typedef**.o:**typedef**.c **typedef**.h

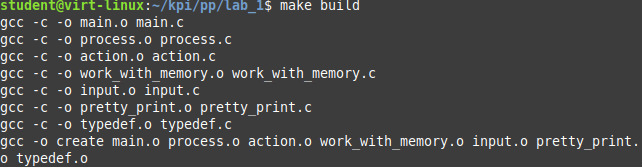
gcc -c -o **typedef**.o **typedef**.c

create:main.o process.o action.o work\_with\_memory.o input.o pretty\_print.o **typedef**.o

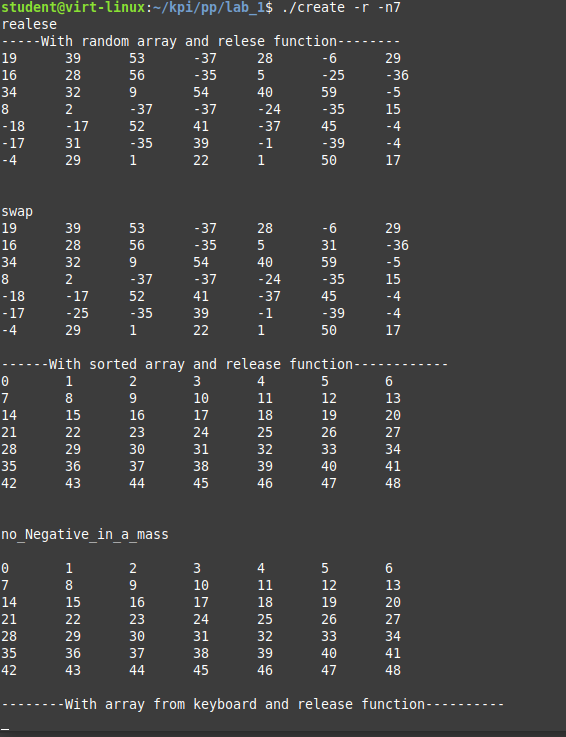
gcc -o create main.o process.o action.o work\_with\_memory.o input.o pretty\_print.o **typedef**.o

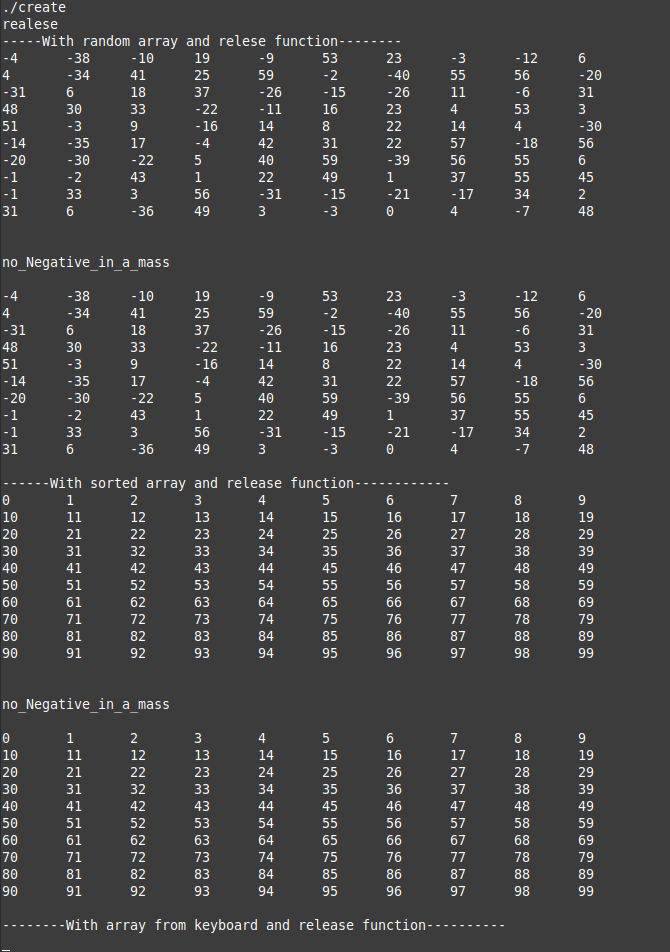
Тести:

Побудова програми:

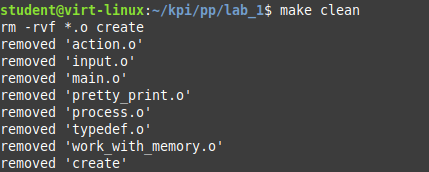


Тест роботи с параметрами:

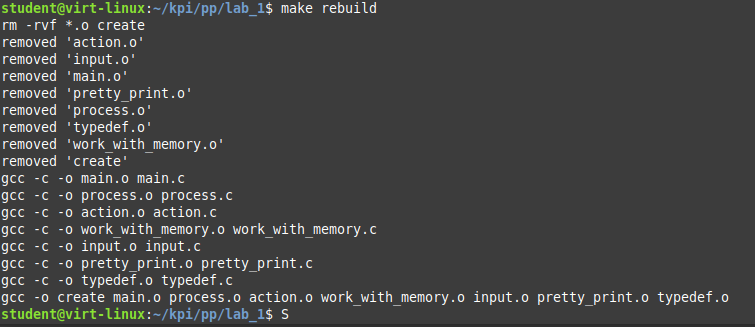


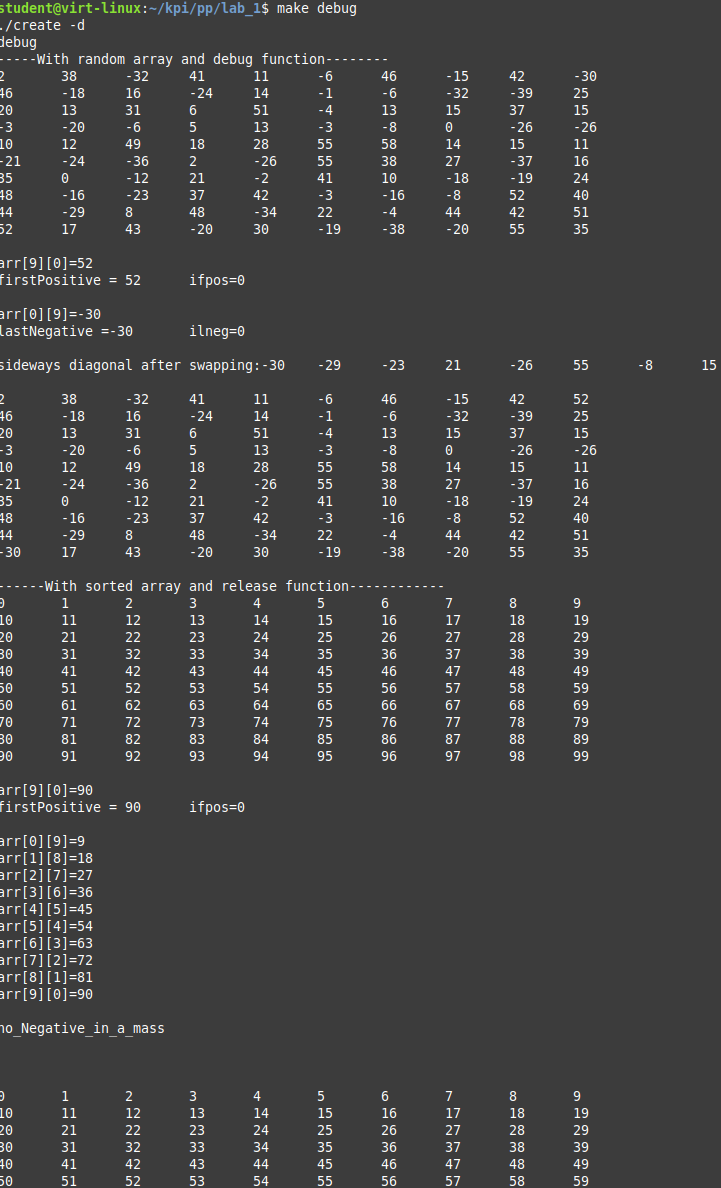
без параметрів: 

очищення:



Перебудова:



Дебаг режим: 

Режим реліз: 