МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 1

з дисципліни “ Основи програмування 2”

тема “Модулі”

Виконав Перевірив

студент I курсу “23” “березня” 2019р.

групи КП-81 викладач

Подлеснюк Богдан Анатолійович Гадиняк Руслан Анатолійович

варіант №20

Київ 2018

### Мета роботи

Навчитись створювати модулі в мові С.  
Навчитись реалізовувати основні колекції елементів різними способами.  
Навчитись розбивати код проекту на декілька файлів і компілювати його.

**Постановка завдання**

Створити у репозиторії progbase2 директорію проекта labs/lab1/

Додати у директорію проекта текстовий файл data.txt, що містить довільні рядки з англійськими словами та реченнями. Кожне речення з нового рядка.

Динамічний список. Реалізувати список рядків на основі динамічного масиву.

Зчитати всі текстові рядки з текстового файлу і додати їх у список (UPD-1 кожен рядок потрібно копіювати на кучу і додавати у список вказівник на початок).

Вивести вміст списку у консоль.

Перенести всі рядки, у яких немає символа '!', на початок списку.

Вивести вміст списку у консоль.

Інтерфейс АТД. Реалізувати стек рядків на основі динамічного масиву

Створити два екземпляри даного типу даних.

Переписати всі елементи непарних позицій списку у перший стек, а всіх парних - у другий.

Вивести вміст обох екземплярів у консоль.

Створити новий список і переписати у нього всі елементи з першого стека, а потім з другого.

Вивести вміст списку у консоль.

Весь код повинен бути розбитим на модулі відповідно до завдання:

main - головний модуль з функцією main()

list - модуль списку (інтерфейс)

stack - модуль даного АТД (інтерфейс)

При найменуванні модулів, типів та операцій обов'язково слідувати рекомендаціям.

Кожен модуль розміщувати у двох файлах (.h i .c). Компілювати проект за допомогою команди (з директорії проекта):

gcc \*.c

При створенні модулів допускається використання відкритих структур, але зовнішній код не повинен напряму доступатися до полів такої структури (перевіряється тимчасовим перенесенням відкритої структури у реалізацію, при чому єдиною помилкою має стати створення екземпляра на стеку).

Контрольні питання

Що таке модуль, інтерфейс, реалізація і зовнішній код?

Що таке інкапсуляція, інтерфейс, реалізація?

Що таке модель, тип даних і абстрактний тип даних?

Назвіть декілька типів поширених колекцій елементів.

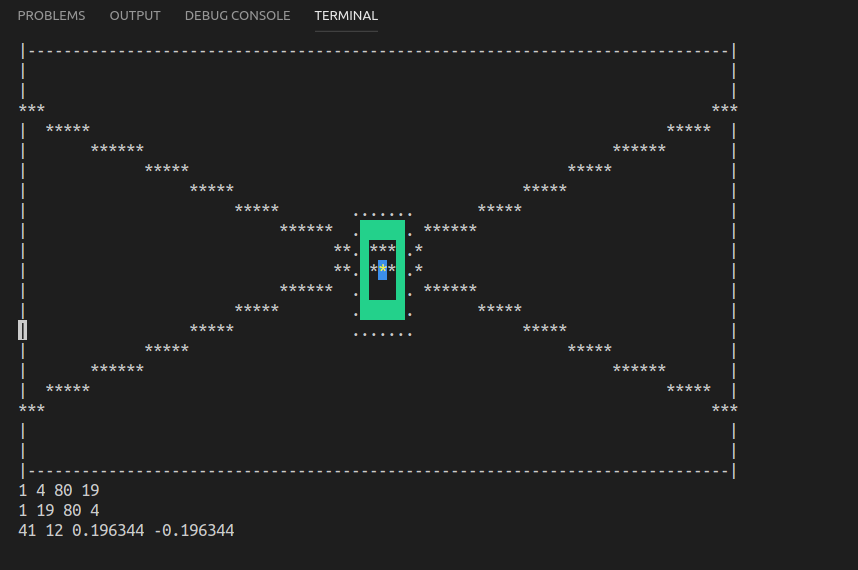
Для чого в С використовується ключове слово typedef?

**Тексти коду програм**

|  |
| --- |
| **main.c** |
| **#include <stdlib.h>**  #include <string.h>  #include <stdio.h>  #include <assert.h>  #include "list.h"  #include "stack.h"  #define ORANGE "\033[0;38;5;214m"  #define RESET "\033[m"  int main() {  List \* plist1 = List\_alloc();  List\_add(plist1, "self1");  List\_add(plist1, "secr");  List\_readStringsFromFile(plist1, "data.txt");  List\_add(plist1, "asdtestMATANSASAT!!!");  //printing  printf(ORANGE);  puts("Raw List:");  printf(RESET);  List\_print(plist1);  List\_sortByExclamation(plist1);  //printing  printf(ORANGE);  puts("----------------Sorted List--------------------");  printf(RESET);  List\_print(plist1);  //printing  printf(ORANGE);  puts("------STACKS------");  printf(RESET);  Stack \* pstack1 = Stack\_alloc(1);  Stack \* pstack2 = Stack\_alloc(1);  for(int i = 0; i < List\_size(plist1); i++) {  if(i % 2 == 0) {  //all odd positions(yes, 0 is odd position)  Stack\_push(pstack1, List\_get(plist1, i));  }  else if(i % 2 == 1) {  Stack\_push(pstack2, List\_get(plist1, i));  }  }  //printing  printf(ORANGE);  puts("STACK1--------------------------");  printf(RESET);  Stack\_print(pstack1);  //printing  printf(ORANGE);  puts("STACK2--------------------------");  printf(RESET);  Stack\_print(pstack2);  List \* plist2 = List\_alloc(1);  int length1 = Stack\_size(pstack1);  for(int i = 0; i < length1; i++) {  int size = maxStringLengthInFile("data.txt");  char strbuf[size];  Stack\_pop(pstack1, strbuf, size);  List\_add(plist2, strbuf);  }  int length2 = Stack\_size(pstack2);  for(int i = 0; i < length2; i++) {  int size = maxStringLengthInFile("data.txt");  char strbuf[size];  Stack\_pop(pstack2, strbuf, size);  List\_add(plist2, strbuf);  }  //printing  printf(ORANGE);  puts("--------------------List2---------------------------------");  printf(RESET);  List\_print(plist2);  //memory free  List\_free(plist1);  List\_free(plist2);  Stack\_free(pstack1);  Stack\_free(pstack2);  return 0;  } |

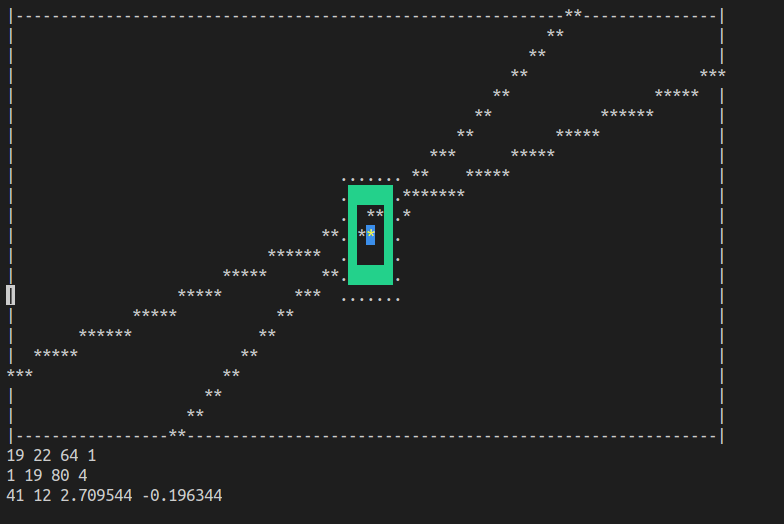
**Приклади результатів**

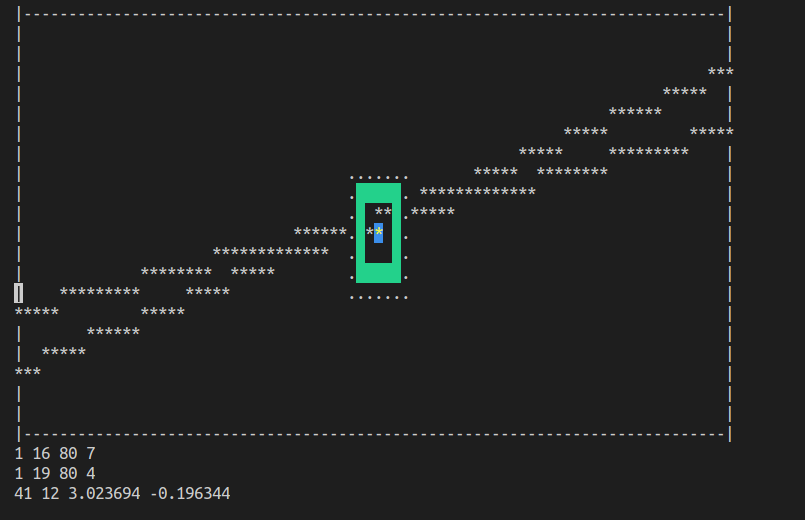
Стандартний вигляд графічної сцени:

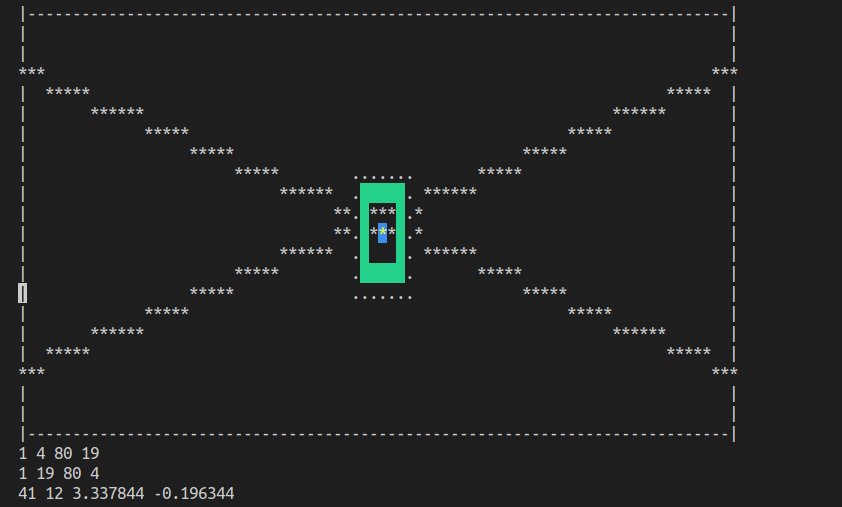


При повороті лінії **а** за годинниковою стрілкою:

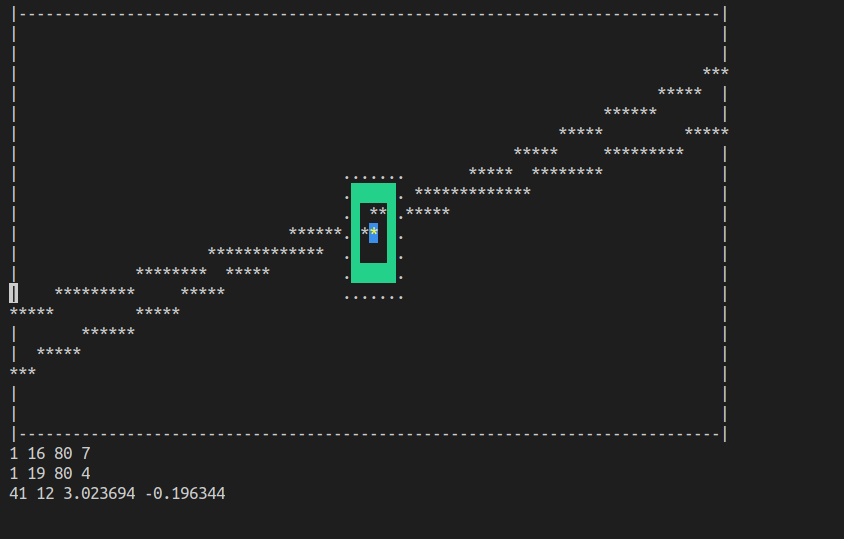


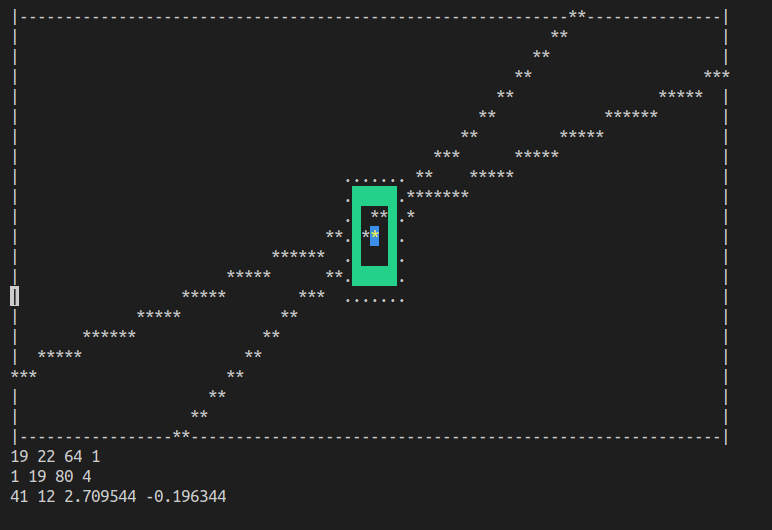






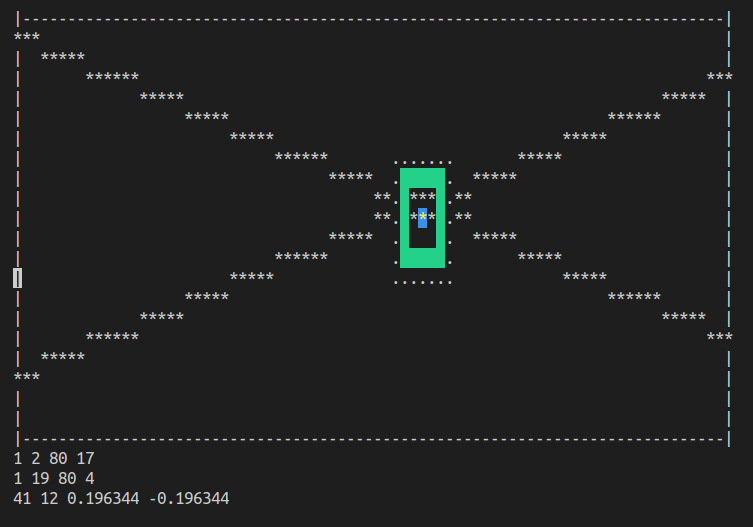
При повороті **а** проти годинникової стрілки:





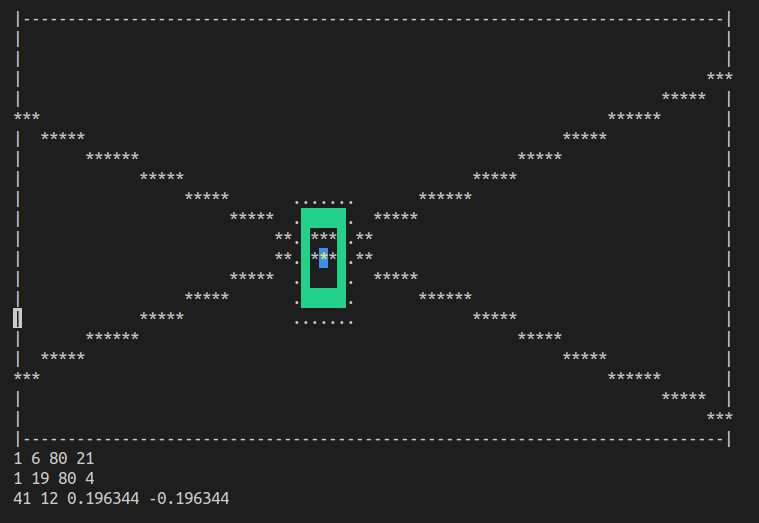


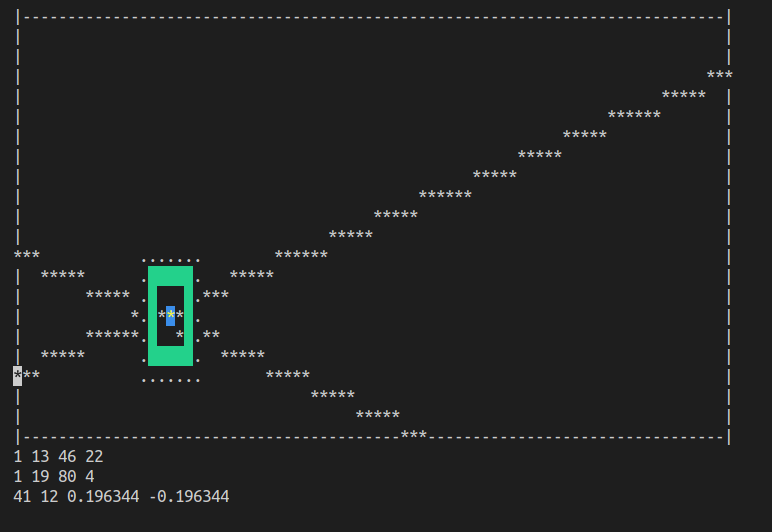
При пересуванні **а** вгору:



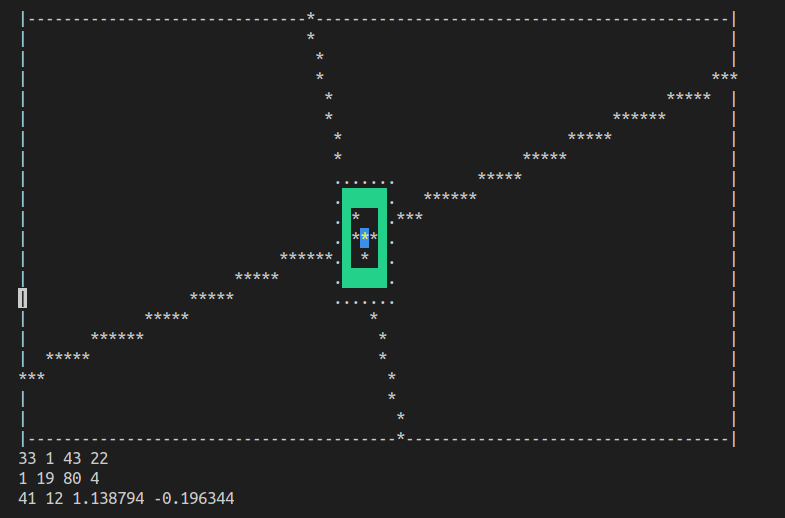


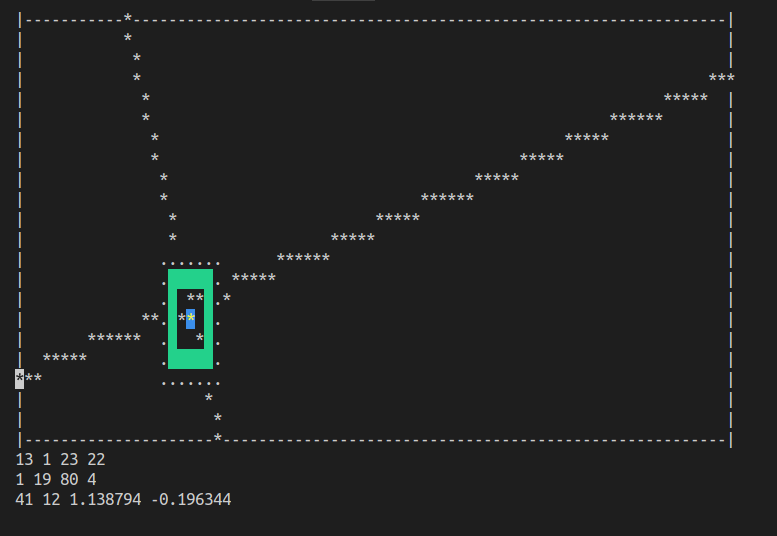
При пересуванні **а** вниз:



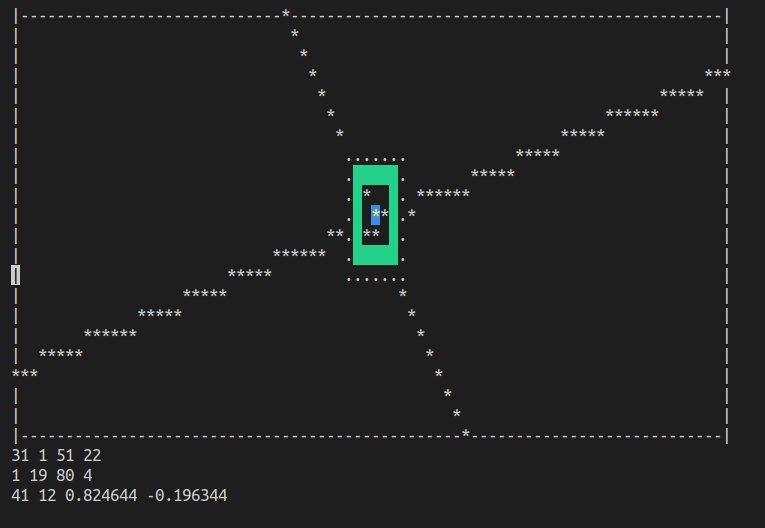


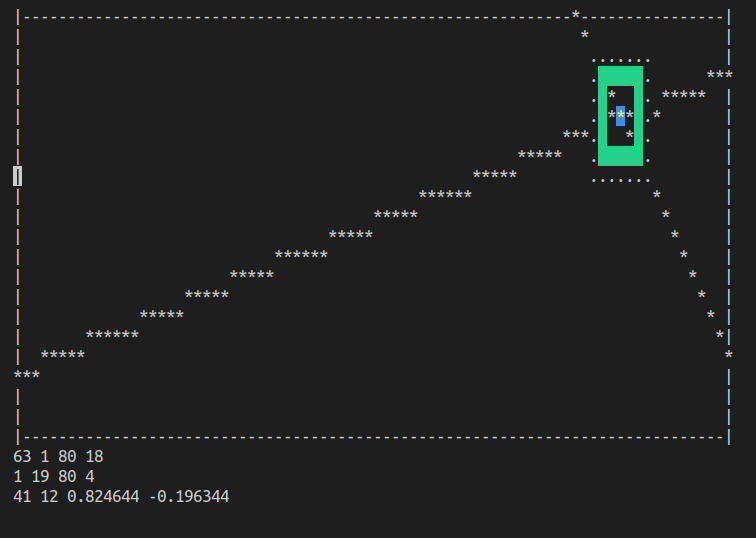
При пересуванні **а** вліво:



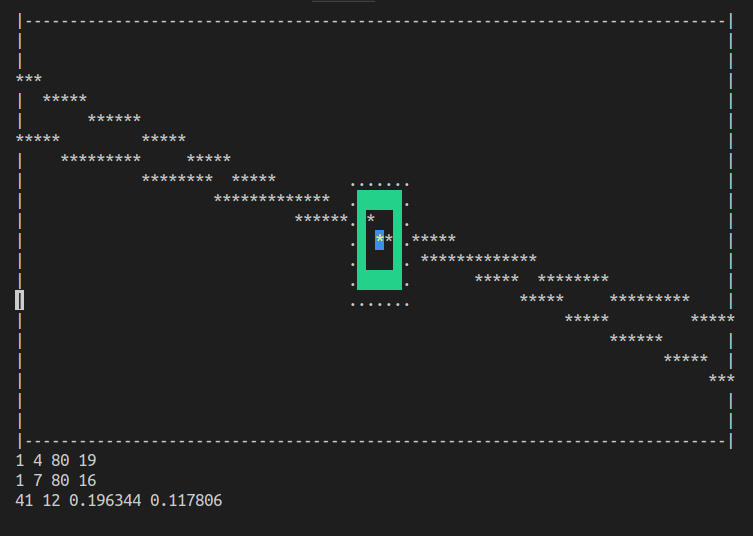


При пересуванні **а** вправо:

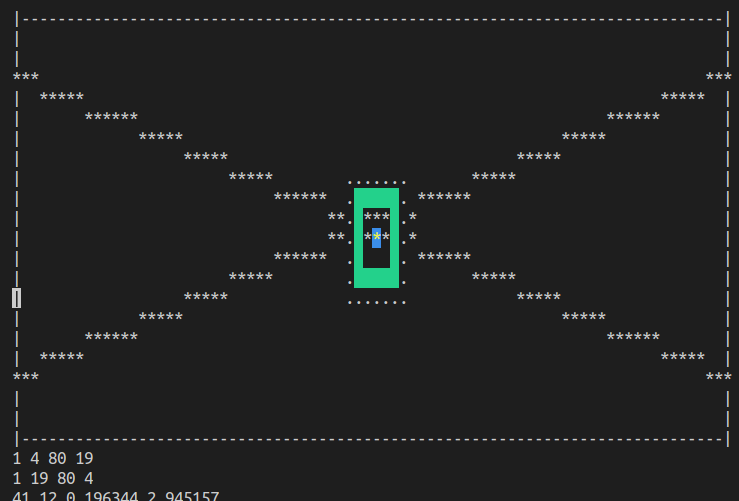




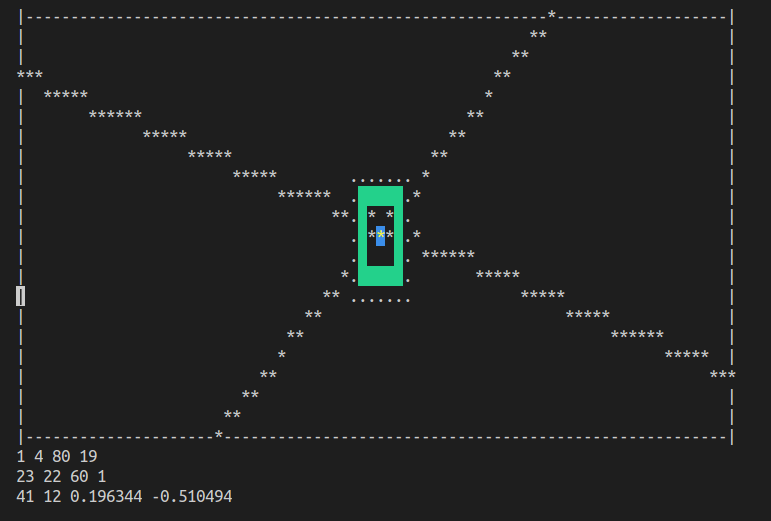
При повороті лінії **б** за годинниковою стрілкою:

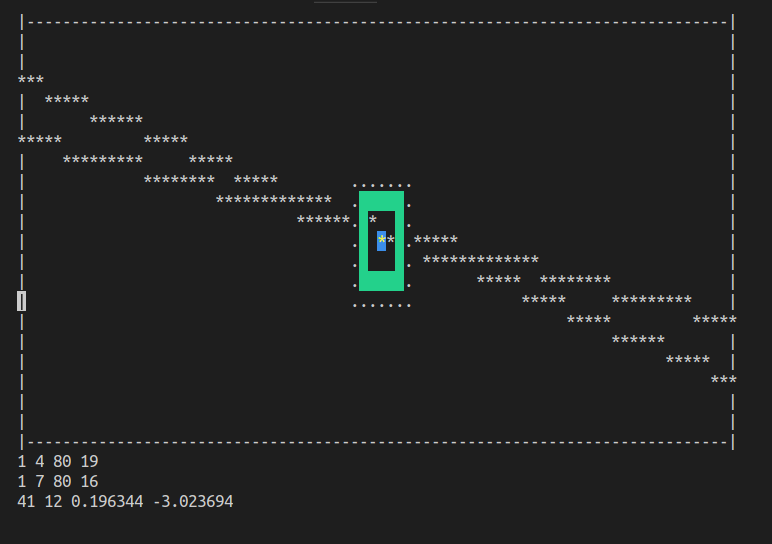




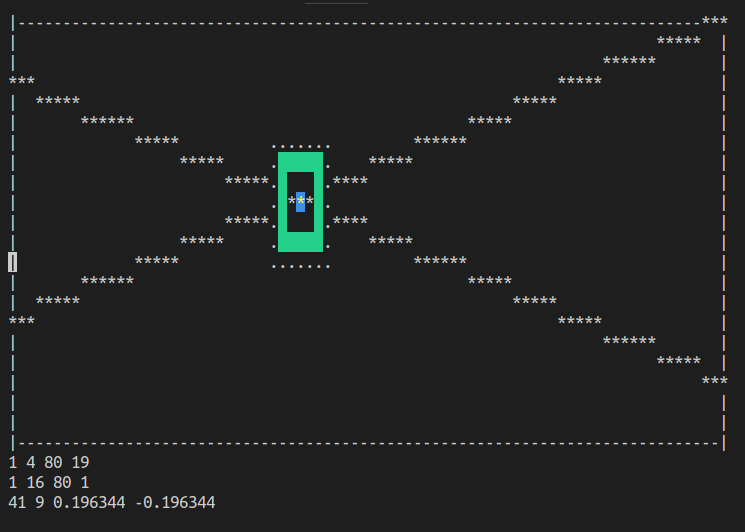


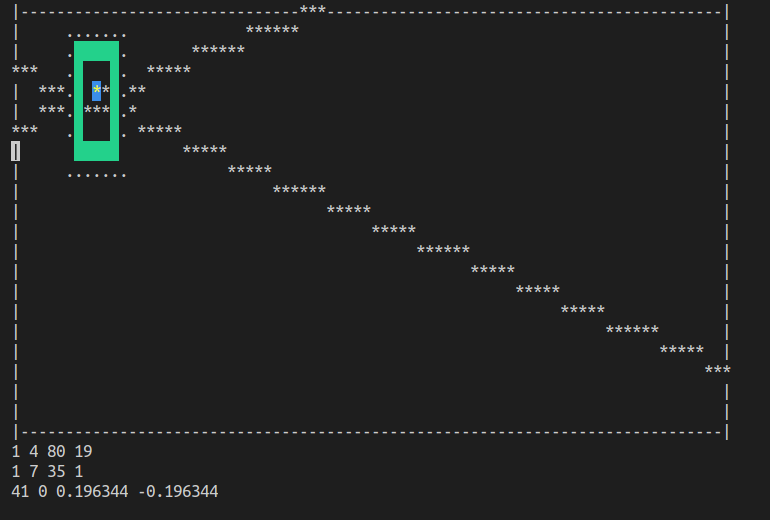
При повороті **б** проти годинникової стрілки:



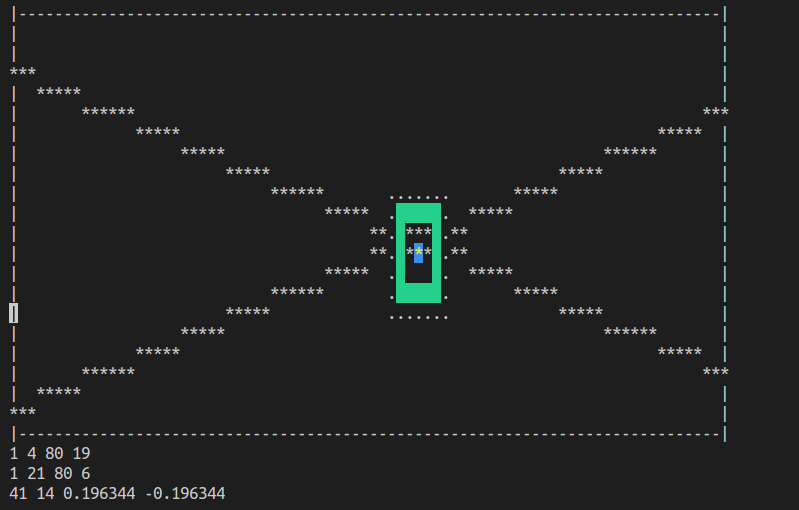


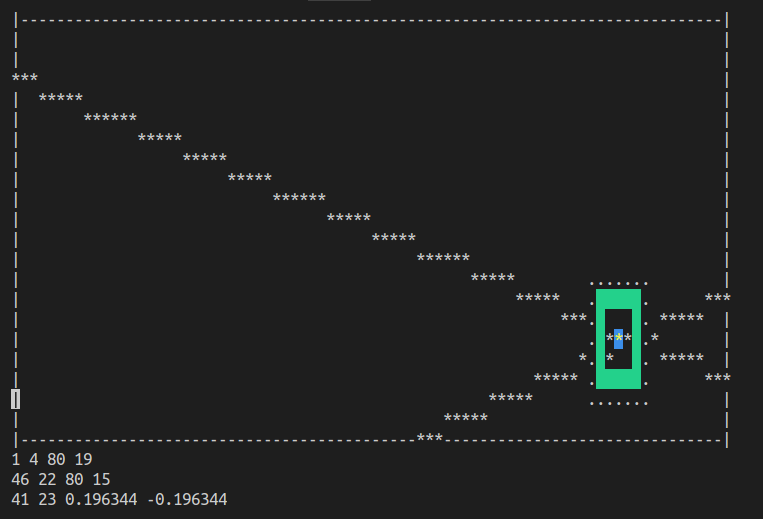
При пересуванні **б** вгору:



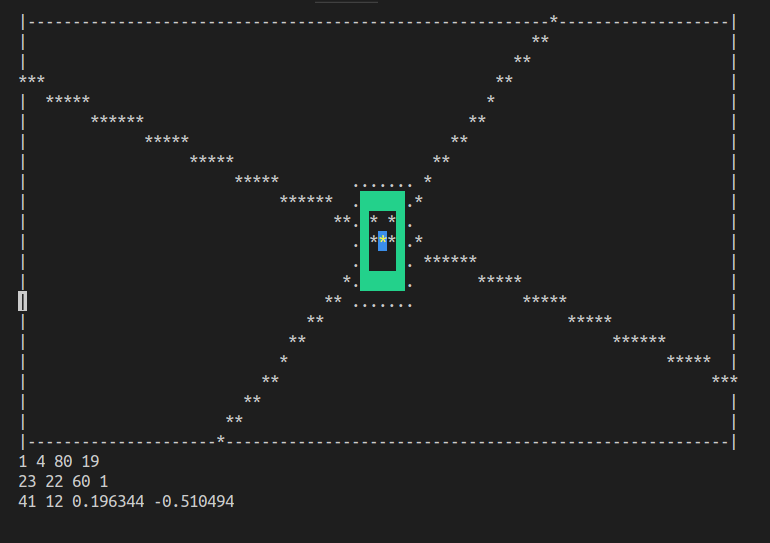


При пересуванні **б** вниз:





При пересуванні **б** вліво:

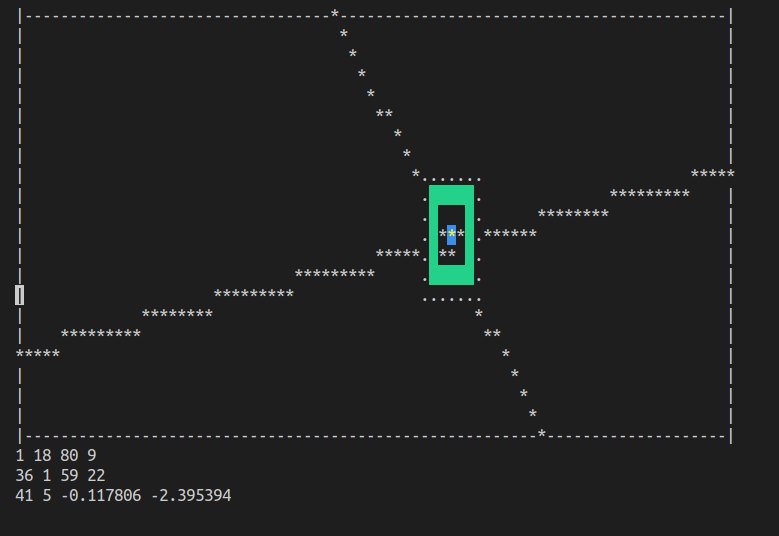


При пересуванні **б** вправо:

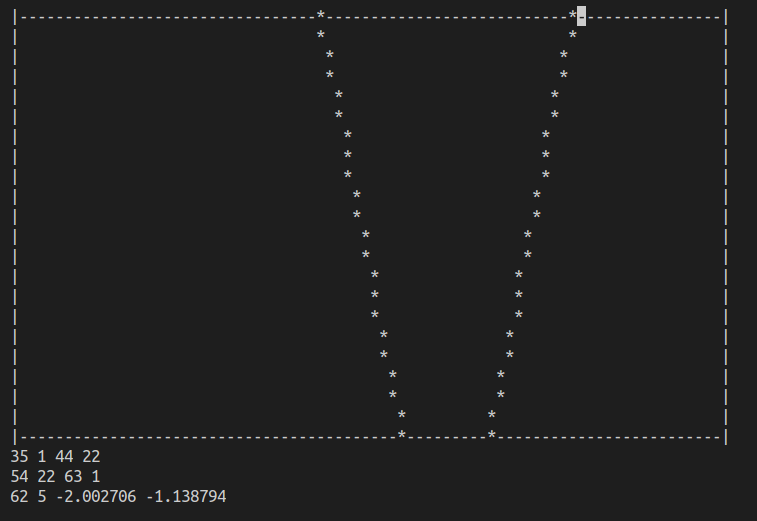


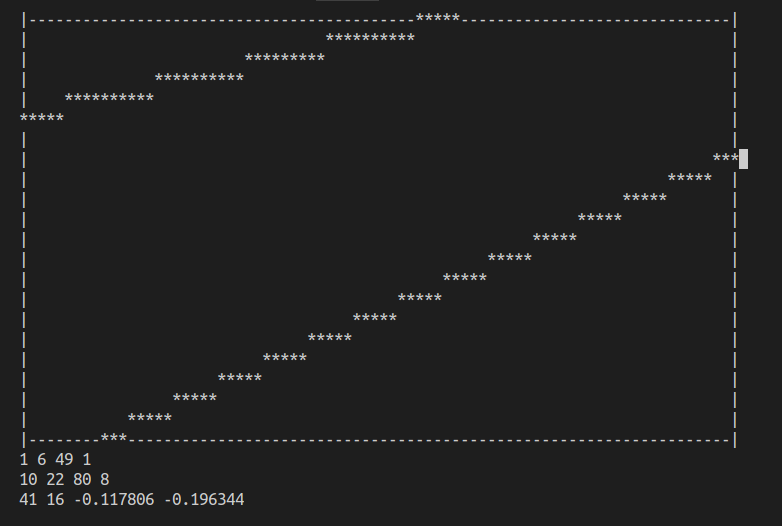
Змішані варіанти зміщень та переворотів:



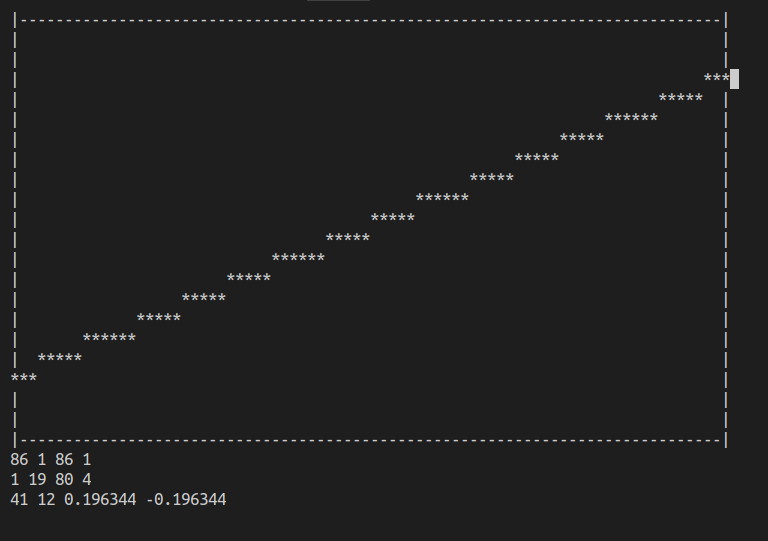


Коли **а** паралельна **б** чи не перетинає її:

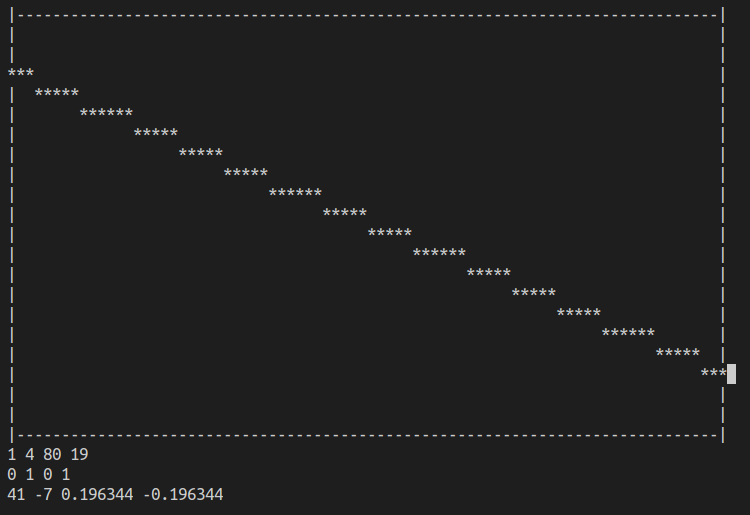




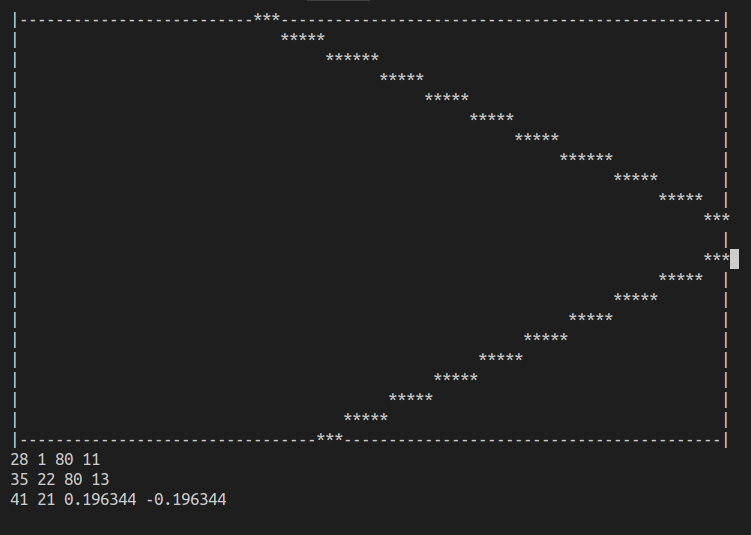
Коли **а** вийшла за поле:



Коли **б** вийшла за поле:



Коли лінії перетинаються за полем:



**Висновки**

Виконавши дану лабораторну роботу, ми навчилися використовувати на практиці різні види циклічних конструкцій та параметризувати алгоритми для роботи із геометричними об'єктами.

В результаті виконання лабораторної роботи для побудови геометричних фігур був застосований цикл for.

Компіляція всього коду відбувалася за допомогою компілятора gcc.