Лабораторна робота №7. Функції

**Автор:** Марков Владислав Андрійович

**Група:** КН-922Б

**Завдання:**

**1.**Переробити програми, що були розроблені під час виконання лабораторних робіт з тем "Масиви" та "Цикли" таким чином, щоб використовувалися функції для обчислення результату.

**2.**Функції повинні задовольняти основну їх причетність - уникати дублювання коду.

Тому, для демонстрації роботи, ваша програма (функція main()) повинна мати можливість викликати розроблену функцію з різними вхідними даними.

**3.**Слід звернути увагу: параметри одного з викликів функції повинні бути згенеровані за допомогою генератора псевдовипадкових чисел random().

**4.**Слід звернути увагу (#2): продемонструвати встановлення вхідних даних через аргументи додатка (параметри командної строки).

Обробити випадок, коли дані не передались - у цьому випадку вони матимуть значення за умовчуванням, обраними розробником.

**Опис програми**

***Функціональне призначення***

Ця програма виконує дві операції.

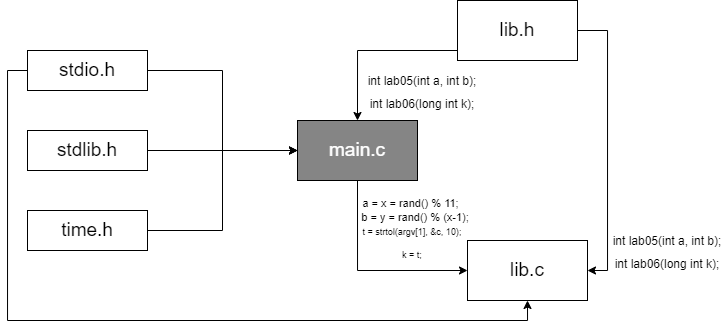
**1.** Обчислює кількість щасливих квитків

* При запуску програми ви повинні ввести номер квитка, від якого хочете почати розрахунок.
* Якщо ви не введете номер квитка, то розрахунок буде здійснено з квитка під номером 1000

**Увага!** Обчислення кількості щасливих квитків можливе лише якщо буде введено чотири розрядне число, якщо буде введено трьох розрядне або п'яти розрядне число розрахунок буде не можливий

**2.**Знаходить найбільше просте число в діапазоні випадкових чисел.

* Програма формує випадкові числа і знаходить між ними найбільше просте число

***Опис логічної структури***

*(Кар. 1) Графічна структура програми*

***Вміст файлу "main.c"***

Головний файл

Це файл, який містить точку входу, виклики функцій lab05, lab06 та значення для аргументів цих функцій.

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

## Головна функція.

Послідовність дій

* Присвоїти значення аргументам argc і argv.

1. *argc - аргумент типу int необхідний для обчислення всіх щасливих квитків.*
2. *argv - масив типу char який зберігатиме в собі значення від якого почнеться перевірка всіх квитків.*

* Створення змінних яким буде надано значення для аргументів функцій link lab05, lab06.

1. *t - зберігає значення за умовчуванням від якого буде починатися перевірка квитків для функції lab06. Воно буде використовуватися якщо не будуть передані аргументи командного рядка.*
2. *x - зберігає рандомне значення, яке є мінімальним числом в діапазоні для функції lab05.*
3. *y - зберігає рандомне значення, яке є максимальним числом в діапазоні для функції lab05.*
4. *c - змінна яка використовується для перевірки аргументів командного рядка.*

* Гненеруємо рандомні числа за допомогою генератора rand() та функції srand(), після чого привласнюємо їх пріменним x і y.
* Потім викликаємо функцію lab05 і присвоюємо її аргументам значення примінних x та y.

*srand((unsigned int)time(NULL));*

*x = rand() % 11;*

*y = rand() % (x-1);*

*lab05(x, y);*

* Робимо перевірку на те, чи були введені аргументи через командний рядок.
* Якщо перевірка була пройдена, то переобразуємо аргумент командного рядка в значення типу int і присвоюємо його примінній t
* Робимо перевірку, чи не менше t мінімального значення квитка 1000.
* Якщо перевірка була пройдена то присвоюємо t аргументам функції lab06
* Якщо одна з перевірок не була пройдена, то присвоюємо аргументам функції lab06 значення за замовчуванням.

*if(t > 1000 && t < 9999)*

*{*

*lab06(t);*

*}else*

*{*

*t= 1000;*

*lab06(t);*

*}*

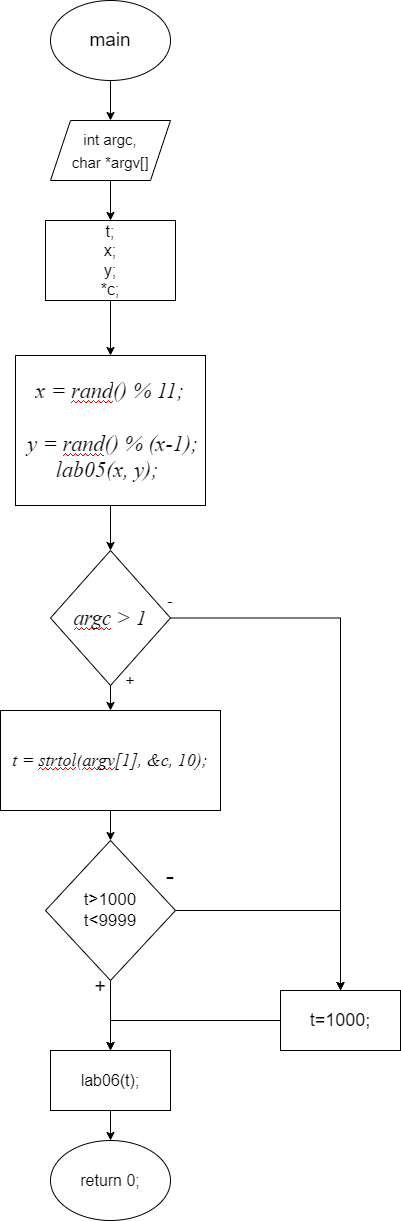
*}else*

*{*

*t= 1000;*

*lab06(t);*

*}*



*(Кар. 2) Схема алгоритму функції main*

***Вміст файлу "lib.c"***

Бібліотечний файл

Цей файл містить реалізацію функцій lab05, lab06.

int lab05(a, b)

Ця функція знаходить найбільше просте число в рандомному діапазоні.

Аргументи

a, b - Діапазон чисел у якому відбувається пошук найбільшого числа.

а - верхнє число діапазону

b - нижня кількість діапазону

Послідовність дій

* Створення змінних i, max.
  1. *i - використовується для перевірки просте число чи ні.*
  2. *max - зберігає найбільше просте число.*
* Запуск циклу для перевірки чисел у встановленому діапазоні.

while(a > b)

* Запуск циклу для того, щоб дізнатися число просте чи ні.

while(b%i != 0){

i++;

}

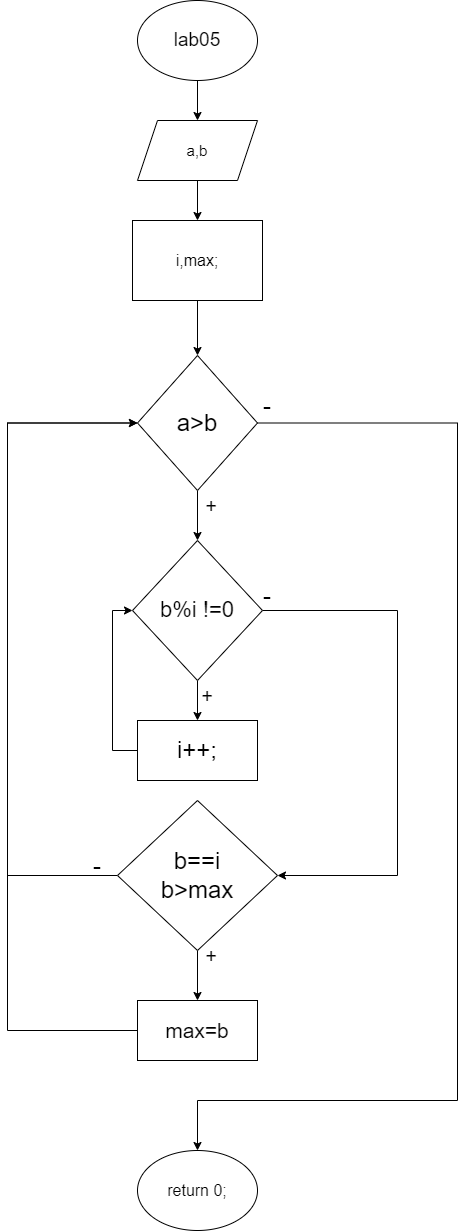
* Робимо перевірку щоб дізнатися число виявилося простим чи ні. Якщо воно просте то перевіряємо чи більше воно числа яке зараз зберігається в змінній max, якщо перевірка була пройдена то присвоюємо змінної max це число.

if(b == i && b > max)

{

max = b;

}



*(Кар. 3) Схема алгоритму функції lab05*

int lab06(k)

Ця функція знаходить усі щасливі квитки в діапазоні від k до 9999.

Аргументи

k - число від якого починається пошук квитків

Послідовність дій

* Створення змінних A[9999],sum1,sum2,n1,n2,n3,n4,n,i.
  + 1. *A - масив в який записуватимемо щасливі квитки.*
    2. *Задаємо змінні sum1 і sum2 в яких буде зберігатися суми першої і другої пари чисел.*
    3. *Змінні n1, n2, n3, n4 зберігатимуть по одному числу з номера квитка.*
    4. *Змінна n – це кількість щасливих квитків.*
* Створення циклу який перевіряє всі квитки від k до 9999.

while( k <= 9999)

* Розбиваємо номер квитка на 4 числа.

n1 = k/1000;

n2 = (k/100)%10;

n3 = (k/10)%10;

n4 = k%10;

* Дізнаємось суму першої пари чисел і другої.

sum1 = n1 + n2;

sum2 = n3 + n4;

* Якщо сума першої пари дорівнює сумі другої, то записуємо цей квиток у масив і додаємо 1 до кількості щасливих квитків(n).

if(sum1 == sum2)

{

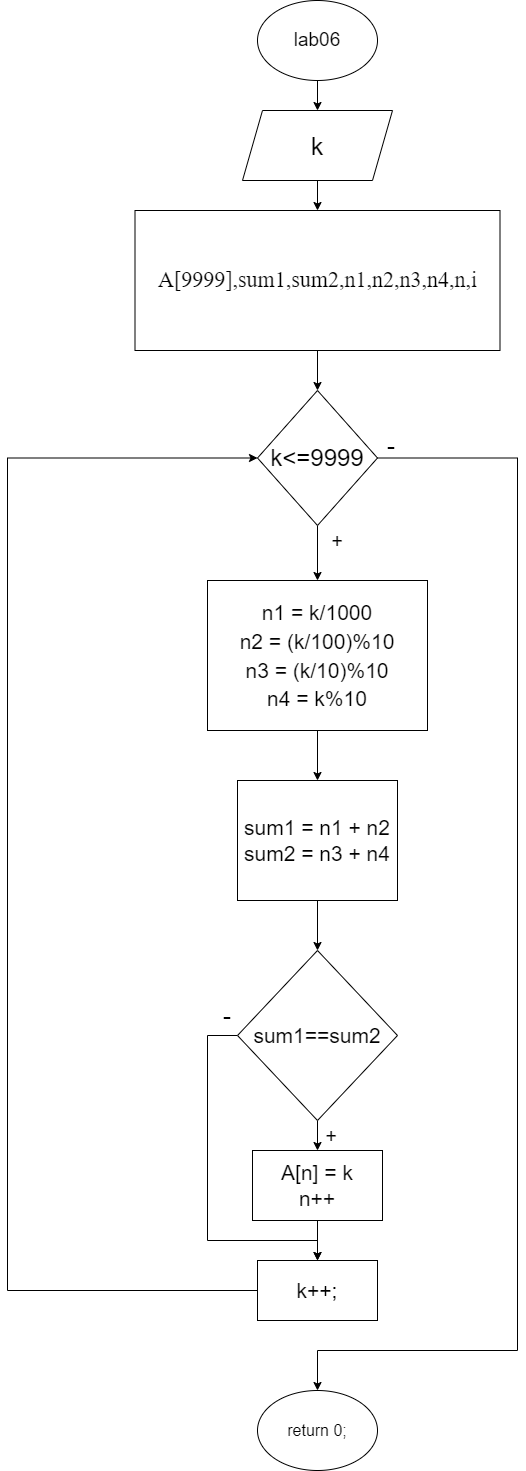
A[n] = k;

n++;

}

* І збільшуємо номер квитка на +1.

k++;



*(Кар. 4) Схема алгоритму функції lab06*

***Вміст файлу "lib.c"***

Бібліотечний файл

Цей файл містить декларацію функцій lab05, lab06.

int lab05(int a, int b);

int lab06(long int k);

***Структура проекту лабораторної роботи:***

├── lab07

├── Makefile

├── README.md

└── src

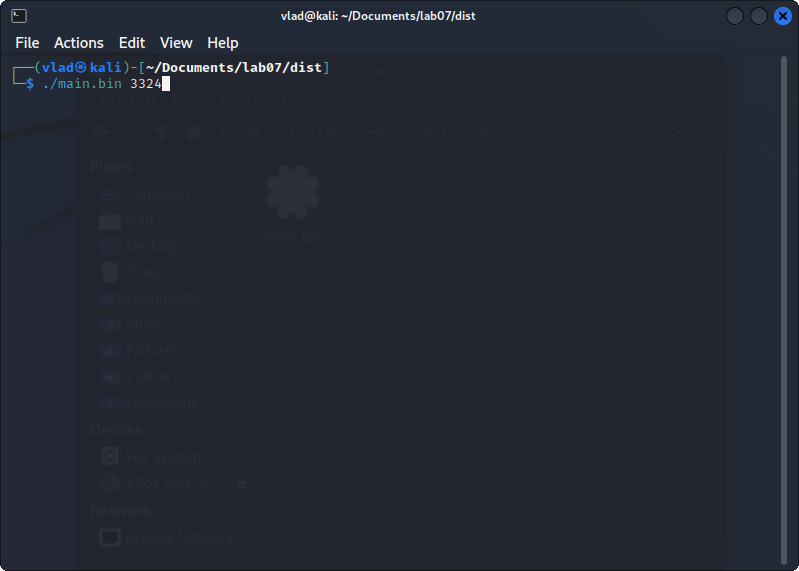
├── lib.c

├── lib.h

└── main.c

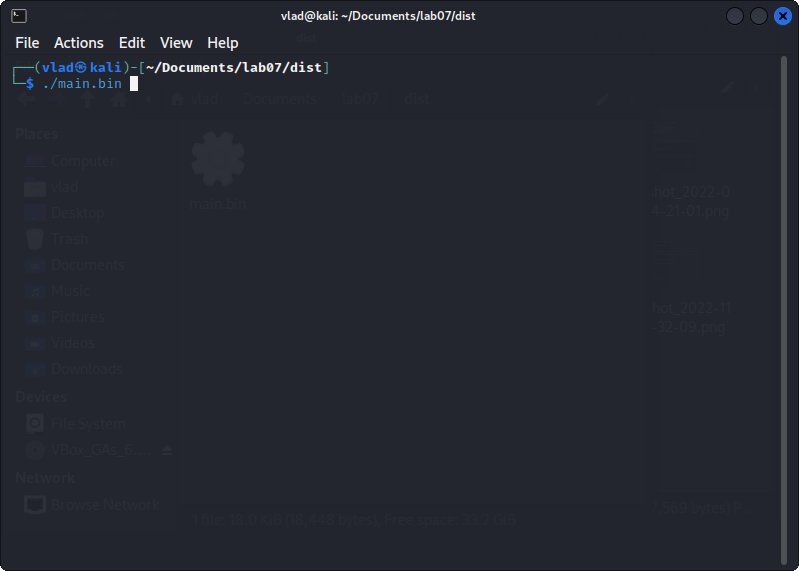
***Варіанти використання***

1. Ви можете використовувати цю програму двома способами. Перший метод - це при запуску двійкового файлу, це вказати номер квитка, з якого буде запущений розрахунок*(Кар 5).* Як описано вище, вам потрібно ввести лише чотирьох розрядне число, якщо введено інше число, програма почне перевіряти квитки з числа 1000.



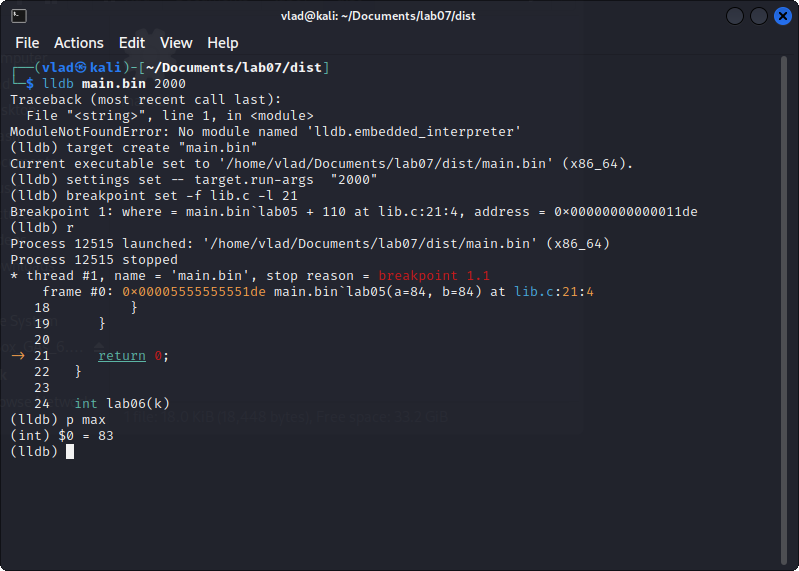
*(Кар. 5)Як правильно користуватися програмою!*

1. Другий метод використання цієї програми - це при запуску двійкового файлу, який не повинен бути дійсним в принципі, якщо ці значення не будуть введені, програма видасть список щасливих квитків від 1000 до 9999. *(Кар. 6)*

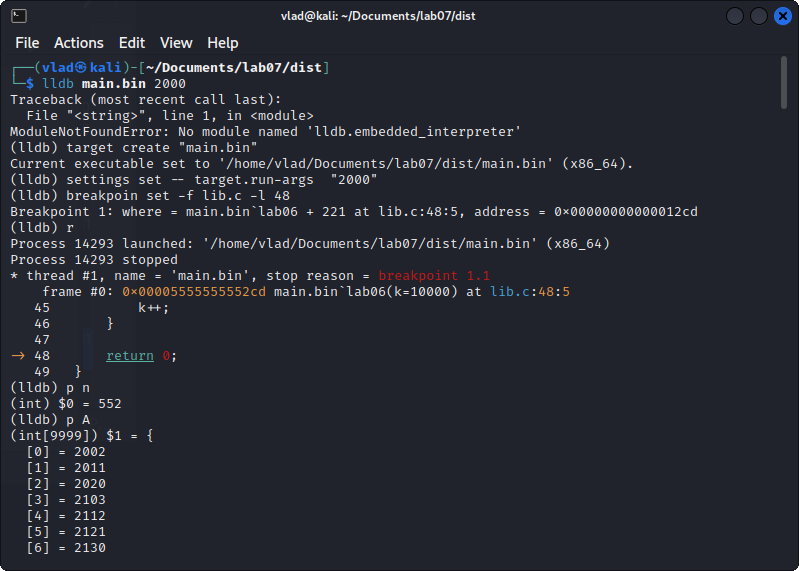


*(Кар. 6)Як правильно користуватися програмою!*

1. Щоб побачити результати роботи програми, вам потрібно завантажити її в LLDB, завантажуючи програму, вкажіть номер квитка. Якщо ви хочете дізнатися найбільше просте число, то для цього вам знадобиться точку зупинки для рядка 21 з "return 0"; У функції lab05, та вивести значення преміального змінно ї max*(Кар. 7)*. Якщо ви хочете дізнатися кількість щасливих квитків або переглянути їх для цього, вам потрібно зробити точку зупинки для рядка 48 з "return 0;" у функції lab06, після того, щоб дізнатися кількість квитків, вам потрібно вивести змінну n, і щоб переглянути щасливі квитки, вам потрібно вивести масив А[]. *(Кар. 8)*



*(Кар. 7) Як дізнатися найбільше просте число!*



*(Кар. 8) Як дізнатися кількість щасливих квитків та їх перегляд!*

Висновки: у цій роботі з було перетворено лабіторні проекти № 5 та № 6 для використання функцій. Було набуто навичок роботи з функціями, їх декларація, реалізація та виклик. Були отримані, також навички роботи з бібліотечними файлами їх зв’язком у виконанні різних дій з функціями та їх зв’язком між собою. Під час тестування програми були отримані результати функції lab05 - це отримання найбільшого числа в діапазоні, і робота функції labb06 - це отримання кількості щасливих квитків та їх перегляду, коли був введений аргумент командного рядка.