

Лабораторна робота №5 Розробка циклічних програм

1 Вимоги

1.1 Розробник

-Носов Микола
-студент групи кіт 1206
-9.11.2020

1.2 Загальне завдання:

Розробити програми, використовуючі 3 типа циклів

1.3 Індивідуальне завдання:

1. Визначити найбільший спільний дільник для двох заданих чисел.
2. Визначити, чи є задане ціле число простим.
3. Визначити, чи є ціле 6-значне число «щасливим» квитком («щасливий квиток» – квиток, в якому сума першої половини чисел номера дорівнює сумі другої половини. Наприклад, білет з номером 102300 є щасливим, бо $1 + 0 + 2 = 3 + 0 + 0$).
4. Визначити, чи є задане число досконалим (якщо воно дорівнює сумі своїх дільників). Наприклад, 6 - досконале число, бо $6 = 1 + 2 + 3$.
5. Без допомоги зовнішніх бібліотек, отримати корінь заданого числа.

2 Опис програми

2.1 Функціональне призначення

Програма призначена для;

- знаходження НОД 2 чисел.
- перевірки на білета на “щастливість”.
- перевірки числа, і визначення, чи є воно простим.
- перевірки числа, і визначення, чи є ідеальним.
- знаххождення квадратного кореня числа.

2.2 Логічна структура

- головна функція main
- структура проєкту (див. Рис.2.2)

```

├── dist
│   └── lab05.bin
├── doc
├── Makefile
├── README.md
├── src
│   └── lab05.c
└── 3 directories, 4 files

```

Рисунок 2.2 - tree

2.3 Важливі фрагменти коду

- обробка виключень (0 та всі та всі знаки, що не є числами)
 - перевірка типу вхідних даних
 - масив для простих дільників числа (див. Рис . 2.4)
 - обробка виключень зі знаком виключень (див Рис 2.5)
- } (див рис 2.3)

```

chislo = chislo1;
if ( chislo == 0){
    printf ( "ERROR\n" ); // проверка на 0 и на знаки
}
if ((chislo1 / (int)chislo) == 1) // Проверить тип вводимых данных

```

Рис 2.3 - умови коректної роботи

```

else if (chislo > 1)
{
    int i = 2;
    int m = 0;
    int mass[20] = {}; // Массив для простых делителей
}

```

Рис 2.4 - масив

```

}

else {
    printf ( "The square root of a negative number is undefined\n"); // Для отрицательных чисел
}

```

Рис 2.5 - виключення для цифр зі знаком (-)

3.0 Визначення результатів

Для обчислення результатів ми використаємо gdb - дебагер, влаштований в утиліту gcc.

Щоб подивитися на результат, ми повинні при компіляції вказати рівень інформації для відладки за допомогою -o, відкомпілювати код, запустити його бінарний файл у відлагоднику та вказати головну функцію, як місце, з якого починати відладку за допомогою команди “b main”, запустити програму та дивитись за результатом її роботи.

Щоб подивитись на значення змінних, використовуємо “info locals”
Значення змінних для першого випадку (див рис 3.1, див коментарі до коду)

```
A = 520;  
B = 260;  
N= 123123;  
a12= 7;  
Bi=6;
```



```
nod = 260  
x2 = 6  
x3 = 6  
n = 0  
j = 1  
a12 = 7  
result = 0  
bi = 6  
pi = 6  
ci = 6  
chislo1 = 0  
chislo = 0  
s = -102 '\232'  
digit = 0  
(gdb) □
```

Рисунок 3.1 - значення змінних у gdb

Значення змінних для другого випадку (див рис 3.2, див коментарі до коду)


A=400;

B=260;

N=123143;

A12=8;

Bi=7;



```
nod = 20
x2 = 8
x3 = 6
n = 0
j = 0
a12 = 7
result = 0
bi = 7
pi = 0
ci = 1
chislo1 = 200
chislo = 200
s = -102 '\232'
digit = 200
```

Рисунок 3.2 - значення змінних у gdb

Значення змінної для 5 задачі див рис 3.3 :

Chislo = 200;



```
10√2 ~ 14.142136
```

Рисунок 3.3 - результат роботи програми

Висновок :

У ході даної лабораторної роботи ми навчились використовувати 3 види циклів.