

# Лабораторна робота №3 Розробка лінійних програм

## 1 Вимоги

### 1.1 Розробник

-Носов Микола  
-студент групи кіт 120б  
-26.10.2020

### 1.2 Загальне завдання:

Розробка лінійних програм без використання введення та виведення змінних, циклів, умов

### 1.3 Індивідуальне завдання:

Виконати 6 лінійних задач з останньої категорії

## 2 Опис програми

### 2.1 Функціональне призначення:

1. Визначення загального опору 3-х паралельно з'єднаних резисторів  
Створюю 4 змінні, r, r1, r2, r3 (Рис 1)  
r - загальний опір  
r1,r2,r3-опір резисторів

```
1 #include <stdio.h>
2
3 //1 Сопротивление резисторов
4 int main()
5 {
6
7
8     float r = 0;
9     int r1 = 1;
10    int r2 = 2;
11    int r3 = 3;
12    r = 1/((1/r1)+(1/r2)+(1/r3));
13
```

(Рис 1)-код задачі

Опис логічної структури:

Задаємо тип даних float(дробові) та записуємо значення опору резисторів

Щоб перевірити роботу нашої програми запускаємо дебагер та дивимось за значенням загального опорору (Рис 2)

```
3 r = 0.545454562
4 r1 = 1
5 r2 = 2
6 r3 = 3
```

(Рис 2)-значення змінних

## 2. Перевернення трицифрового цілого числа(Рис 3)


```
15
16 //2 Перевернуть число
17     int n, reverse, q, v, g;
18     n=522;
19     q=n%10;
20     v=n/10%10;
21     g=n/100%10;
22     reverse=q*100+v*10+g;
23
24
```

(Рис 3)-фрагмент коду

### Опис логічної структури:

Задаю змінні int формату, та розбиваю число на сотні, десятки, одиниці

Запускаю відкомпільований файл у gdb та дивлюсь на роботу програми(Рис4)



```
n = 522
reverse = 225
q = 2
v = 2
g = 5
```

(Рис 4)-значення змінних

## 3. Зведення в ступінь з обмеженою кількістю операцій Створюю змінну a,a4,a6,a8,a10 та ввожу в ступінь (Рис 5)

```
//3 Возвести в степень за 2,3,4 действия
```

```
int a, a4, a6, a8, a10;
a=2;
a4=a*a;
a4=a4*a4;
a6=a*a*a;
a6=a6*a6;
a8=a*a;
a8=a8*a8;
a8=a8*a8;
a10=a4*a;
a10=a10*a10;
```

(Рис 5)-фрагмент коду

### Опис логічної структури:

Використовую int формат для змінних та записую операції отримання результату

За допомогою відладчика демонструю результат (Рис 6)

```
a = 2
a4 = 16
a6 = 64
a8 = 256
a10 = 1024
```

(Рис 6)-значення змінних

4. Переведення числа із заданої системи числення у десятирічну  
Зауваження: програма не створена для переведення із 16-річної системи числення (Рис 7)

```
//4 Перевести в 10-ую систему исчислений
int chislo,p, a4,b4,c4,d4,chislo10;
chislo = 1337;
p = 8;
a4 = (chislo/1000)*(p*p*p);
b4 = ((chislo/100)%10)*(p*p);
c4 = ((chislo/10)%10)*p;
d4 = chislo%10;
chislo10 = a4+b4+c4+d4;
```

(Рис 7)-фрагмент коду

#### Опис логічної структури:

Необхідно розбити число на цифри, а потім домножити на число еквівалентне заданій системі числення у ступені, який дорвнює порядковому місцю цифри у числі, починаючи від 0 та записати суму цих чисел

Відкриваємо дебагер та дивимося на значення змінних(Рис 8)

```
chislo = 1337
p = 8
a4 = 512
b4 = 192
c4 = 24
d4 = 7
chislo10 = 735
```

(Рис 8)-значення змінних

5. Підрахування у скільки разів перша цифра числа є більшою за останню(Рис 9)

Зауваження: треба округлити значення до 2 цифт після крапки

```

0
1
2
3 //5 Поделить первую цифру числа на последнюю
4
5
6
7     int x = 123;
8     int b1 = x/100;
9     int c1 = x%10;
0     int a10=(int)((float)b1/(float)c1 * 100);
1     float result = (float)a10/100;

```

(Рис 9)-фрагмент коду

#### Опис логічної структури:

Достаємо із числа 1 та 3 цифри за допомогою математичних операцій.

Ділимо 1 цифру на 3, для того щоб округлити домножаємо отримане число на 100, переводимо у int, потім знову у float та ділимо на 100

Відкриваємо дебагер та дивимось результат (Рис 10)

```

x = 123
b1 = 1
c1 = 3
a10 = 33
result = 0.330000013

```

(Рис 10)-значення змінних

#### 6. Підрахування суми арифметичної прогресії

Створюємо змінну a1 для початку арифметичної прогресії та an для кінця  
За формулою підраховуємо суму(Рис 11)

```

//6 Сумма арифметической прогрессии
float a1, an , c6, suma;
a1 = 10;
an = 15;
c6 = (an-a1)+1;
suma = ((a1+an)/2)*c6;

```

(Рис 11)

**Опис логічної структури:** за допомогою формули обчислюємо сумму всіх членів арифметичної прогресії із шагом в 1. Дивимось результат у дебагері.  
(рис 12)

```

a1 = 10
an = 15
c6 = 6
suma = 75

```

(Рис 12)-значення змінних

Структура проєкту (Рис 13):

```
*
├── dist
│   └── lab03.bin
├── doc
│   ├── lab03.docx
│   └── lab03.pdf
├── Makefile
├── README.md
├── src
│   └── lab03.c
└──
```

3 directories, 6 files

(Рис 13)-команда tree

**Висновок:**у рамках данної лабораторної роботи ми навчилися створювати лінійні програми, а також познайомились з новим дебагером gdb