

МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА
ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту



Лабораторна робота №1
З курсу “Алгоритмізація та програмування”

Виконав:
ст.гр. КН-110
Дойков Вадим

Тема: "Знайомство з С. Виконання програми простої структури"

Мета: Знайомство з середовищем програмування, створення, відлагодження й виконання простої програми, що містить ввід/вивід інформації й найпростіші обчислення.

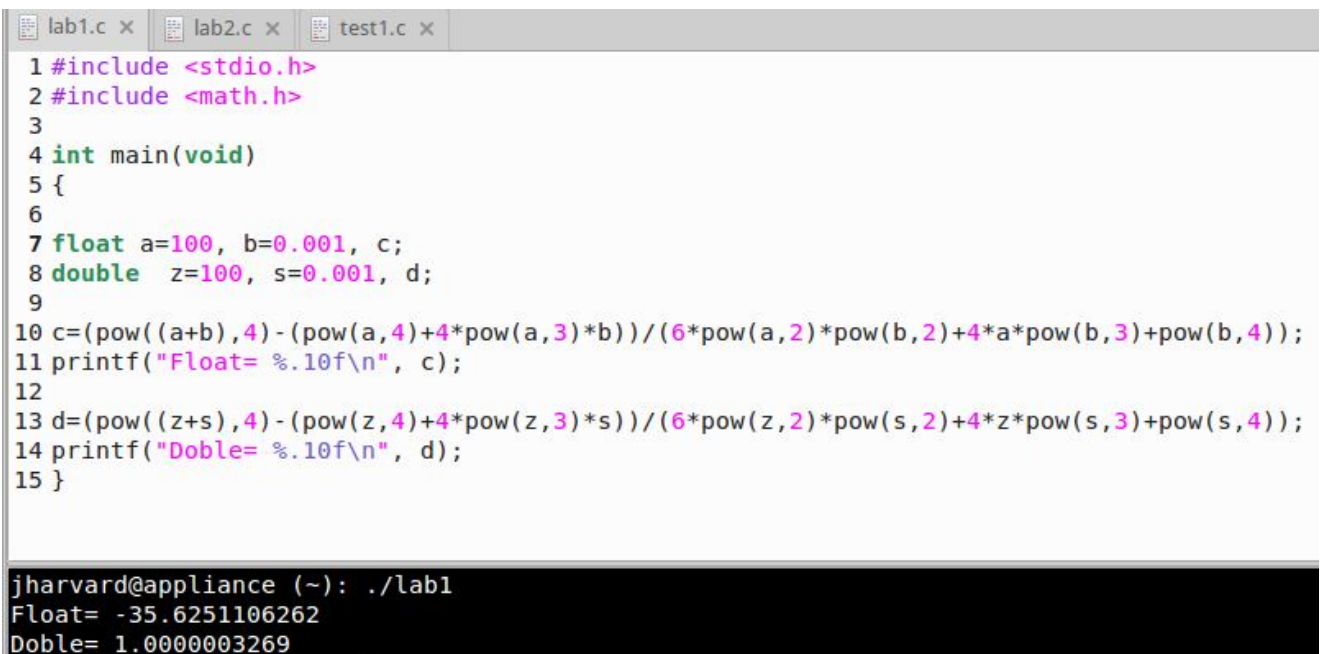
Постановка завдання

1. Обчислити значення виразу при різних дійсних типах даних (float й double).
Обчислення варто виконувати з використанням проміжних змінних.
Порівняти й пояснити отримані результати.
2. Обчислити значення виразів. Пояснити отримані результати.

Завдання 1.

$$\text{Обчислити : } \frac{(a+b)^4 - (a^4 + 4a^3b)}{6a^2b^2 + 4ab^3 + b^4},$$

при a=100, b=0.001



```
lab1.c x lab2.c x test1.c x
1 #include <stdio.h>
2 #include <math.h>
3
4 int main(void)
5 {
6
7 float a=100, b=0.001, c;
8 double z=100, s=0.001, d;
9
10 c=(pow((a+b),4) - (pow(a,4)+4*pow(a,3)*b))/(6*pow(a,2)*pow(b,2)+4*a*pow(b,3)+pow(b,4));
11 printf("Float= %.10f\n", c);
12
13 d=(pow((z+s),4) - (pow(z,4)+4*pow(z,3)*s))/(6*pow(z,2)*pow(s,2)+4*z*pow(s,3)+pow(s,4));
14 printf("Doble= %.10f\n", d);
15 }

jharvard@appliance (~): ./lab1
Float= -35.6251106262
Doble= 1.0000003269
```

printf(" ") та math.h (команда pow(a,b), яка підносить число або вираз до степеня). Також я задіяв різні типи змінних для даних: float(одинарна точність) та double(подвійна точність).

Виконавши програму, отримаємо такий результат:

Float= -35.6251106262

Double= 1.0000003269

Результати різні, тому що різні типи змінних мають різну точність у обчисленні.

Завдання 2.

Обчислити значення виразів:

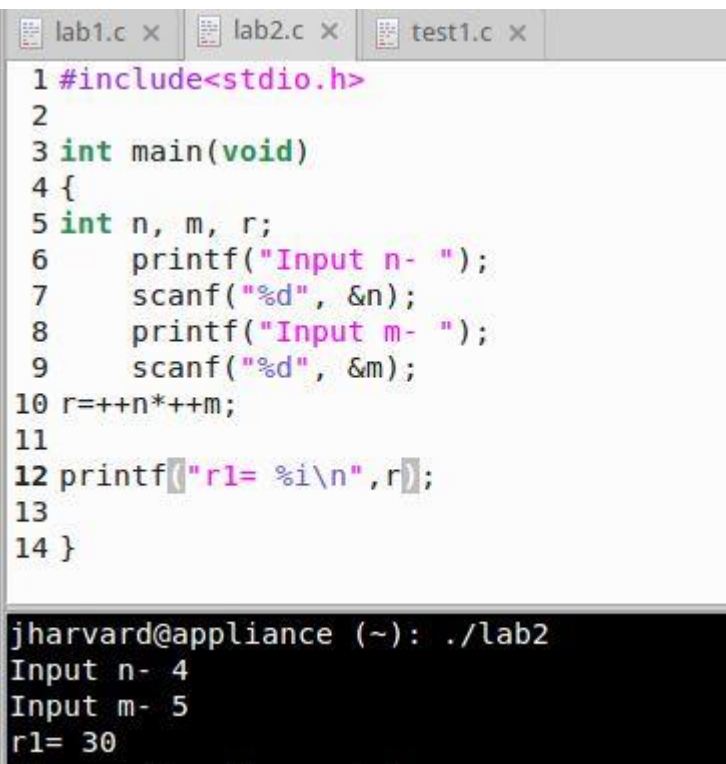
1) $++n*++m$

2) $m++<n$

3) $n++>m$

В першому виразі значення n та m спочатку збільшується на 1, а вже потім перемножуються. В другому та третьому спочатку виконується порівняння змінних, після чого вони збільшуються на 1.

1.



```
lab1.c x lab2.c x test1.c x
1 #include<stdio.h>
2
3 int main(void)
4 {
5     int n, m, r;
6     printf("Input n- ");
7     scanf("%d", &n);
8     printf("Input m- ");
9     scanf("%d", &m);
10    r=++n*++m;
11
12    printf("r1= %i\n", r);
13
14 }
```

```
jharvard@appliance (~): ./lab2
Input n- 4
Input m- 5
r1= 30
```

Ввівши $n=4$ та $m=5$, програма збільшила значення на 1, тобто стало $n=5$, $m=6$, після чого перемножила їх.

В результаті отримаємо:

$r=30$.

2.

```
lab1.c x lab2.c x test1.c x
1 #include<stdio.h>
2
3 int main(void)
4 {
5     int n, m, r1, r2;
6     printf("Input n- ");
7     scanf("%d", &n);
8     printf("Input m- ");
9     scanf("%d", &m);
10    r1=m++<n;
11
12    printf("r1= %i\n",r1);
13    r2=m<n;
14    printf("r2= %i\n",r2);
15
16 }
```

```
jharvard@appliance (~): ./lab2
Input n- 6
Input m- 5
r1= 1
r2= 0
jharvard@appliance (~):
```

В цій програмі порівнюються дві змінні: n та m

Якщо $n > m$, то відповідь відповідає умові і дорівнює 1 (True).

Якщо $n < m$, то відповідь не відповідає умові і дорівнює 0 (False).

Після порівняння, програма збільшила m на 1.

3.

```
lab1.c x lab2.c x test1.c x
1 #include<stdio.h>
2
3 int main(void)
4 {
5     int n, m, r;
6     printf("Input n- ");
7     scanf("%d", &n);
8     printf("Input m- ");
9     scanf("%d", &m);
10    r=n++>m;
11
12    printf("r2= %i\n",r);
13
14 }
```

```
jharvard@appliance (~): ./lab2
Input n- 6
Input m- 5
r2= 1
```

Третя програма працює за аналогічним принципом, що і 2.

Висновок: я ознайомився з середовищем програмування C, навчився використовувати стандартні бібліотеки, налагоджувати й виконувати прості програми, що містять ввід/вивід інформації й найпростіші обчислення.