Лабораторная работа №2. Анализ типов данных. Работа с числовыми значениями.

Часть I. Анализ типов данных

Задание 1: Разработайте приложение, позволяющее определить количество байт, занимаемых в памяти, а также диапазон значений целочисленных типов.

1. Создайте новый проект **Example1** в решении **Лабораторная работа 2**;
2. Объявите следующие целочисленные переменные:

```
int size;          //количество байт в типе
long long count;   //количество значений в типе
long int min, max; //границы диапазона значений
```

3. Определите константу для представления количества бит в одном байте:

```
#define BYTE 8
```

4. Определите и выведите количество байт, отводимых на хранение переменной символьного типа:

```
size=sizeof(char);
cout<<"Размер типа: "<<size<<" байт"<<endl;
```

5. Рассчитайте и выведите на экран количество возможных значений типа char, используя формулу:

$$количество\ возможных\ значений = 2^{количество\ бит}$$

```
count=pow(2.,size*BYTE);
cout<<"Количество значений: "<<count<<endl;
```

6. Определите диапазон значений символьного типа:

```
min=-count/2;    //наименьшее значение
max=count/2-1;   //наибольшее значение
cout<<"Диапазон значений: "<<min<<"-"<<max<<endl;
```

7. **Самостоятельно** выполните расчеты для других типов данных и заполните таблицу 1:

Таблица 1. Целые типы данных

Тип данных	Размер (байт)	Количество значений	Минимальное значение	Максимальное значение
char				
unsigned char				
short				
unsigned short				
int				
unsigned int				
long long				

Проанализируйте полученные данные и сделайте вывод.

 © Екатерина Бурова, 2018.

Задание 2: Реализуйте приложение, демонстрирующее работу неявного преобразования типов данных.

1. Добавьте в текущее решение проект **Example2**.
2. Объявите константу **NUM**, определяющую номер вашего варианта (целое число в диапазоне от 1 до 10).
3. Объявите следующие переменные различных типов:

```
char c=char(NUM);
short sh;
int i;
float f;
double d;
```

4. Символьной переменной присвойте код символа, соответствующий вашему варианту, и выведите полученное значение:

```
c=char(NUM);
cout<<"Значение символьной переменной: "<<c<<endl;
```

5. Присвойте переменной типа короткое целое значение символьной переменной и выведите результат на экран:

```
sh=c;
cout<<"Значение переменной типа короткое целое: "<<sh<<endl;
```

6. Самостоятельно выполните преобразование остальных переменных и заполните таблицу 2:

Таблица 2. Преобразование типов от меньшего к большему

Номер варианта		
Исходный тип	**Целевой тип**	**Полученное значение**
char	short	
short	int	
int	float	
float	double	

Проанализируйте полученные данные и сделайте вывод.

7. Значение переменной типа double увеличьте на 1,5.
8. Выполните обратные преобразования типов и заполните таблицу 3:

Таблица 3. Преобразование типов от большего к меньшему

Номер варианта		
Исходный тип	**Целевой тип**	**Полученное значение**
double	float	
float	int	
int	short	
short	char	

Проанализируйте полученные данные и сделайте вывод.

 © Екатерина Бурова, 2018.

Задание 3: Разработайте приложение, демонстрирующее работу с указателями.

1. Объявите целочисленные переменные и присвойте переменной **a** номер вашего варианта, а переменной **b** – ваш возраст.
2. Объявите два указателя на тип int. Указателю **aPtr** присвойте адрес переменной a, **bPtr** – b:

```
aPtr=&a;
bPtr=&b;
```

3. Выведите на печать сообщение «Задание 1» и текущие значения переменных и указателей:

```
cout<<"Задание 1"<<endl;
cout<<"Значение переменной a: "<<a<<endl;
cout<<"Значение переменной b: "<<b<<endl;
cout<<"aPtr ссылается на адрес: "<<aPtr<<endl;
cout<<"bPtr ссылается на адрес: "<<bPtr<<endl;
```

4. Измените значения переменных **a** и **b**, используя косвенную адресацию:

```
*aPtr=a*2;
*bPtr=a+b;
```

5. Выведите на печать сообщение «Задание 2» и текущие значения переменных и указателей.
6. Присвойте переменным **a** и **b** новые значения:

```
a=*bPtr:
b=*aPtr+10;
```

7. Выведите на печать сообщение «Задание 3» и текущие значения переменных и указателей.
8. Измените значение первого указателя:

```
aPtr=bPtr;
```

9. Выведите на печать сообщение «Задание 4» и текущие значения переменных и указателей.
10. Запустите программу и заполните таблицу 4 значениями, полученными в результате ее работы:

Таблица 4. Работа с указателями

№ задания	Значения переменных			
	a	b	aPtr	bPtr
1				
2				
3				
4				

 © Екатерина Бурова, 2018.

Часть II. Работа с числовыми значениями

Задание 4: Разработайте приложение, находящее решение уравнения $x^3 - 3x^2 + 3 = 0$ на промежутке [0;1,5] методом половинного деления.

Метод дихотомии имеет свое название от древнегреческого слова, которое в переводе означает деление надвое. Именно поэтому данные метод имеет еще второе название: метод половинного деления или бисекций. Например, задана какая-то неопределенная функция **f(x)**, имеющая на интервале **[a, b]** только одно решение.

Суть метода состоит в том, что заданный отрезок делится пополам и выясняется, в какой половине находится корень. Это можно сделать проверкой значений функции на концах каждой половины. Там, где лежит корень, значения непрерывной функции имеют разные знаки. После этого выбранная половина отрезка снова делится пополам и все продолжается до тех пор, пока не будет достигнута нужная точность. Таким образом, мы получаем приближенное значение корня уравнения.

1. Добавьте в решение проект **Example4**.
2. Объявите следующие переменные:

```
double a = 0.0, b = 1.5, c, y, eps = 0.01;
//a, b и c – начало, конец и середина интервала соответственно,
//y – значение функции, eps – заданная точность (приближение)

int count=0; //количество итераций
```

3. Опишите алгоритм вычисления корня уравнения:

```
do
{
   count++;                        //увеличиваем счетчик итераций
   c = 0.5 * (a+b);                //вычисляем середину интервала
   y = pow(c,3) - 3*pow(c,2) + 3; //вычисляем значение функции в этой точке

   if (fabs(y) < eps) //если текущее значение функции меньше заданной погрешности
      break;          //решение найдено - выходим из цикла

   //если на концах отрезка функция имеет разные знаки, значит, корень здесь
   if ((pow(a,3)- 3*pow(a,2)+ 3)*y < 0.0)
      b = c; //переносим точку b в точку c
   else      //в противном случае
      a = c; //переносим точку a в точку c
} while (fabs (b-a) >= eps);
//цикл выполняется, пока длина отрезка [a, b] больше заданной точности
```

4. Выведите на экран итоговое значение переменной с как корень уравнения и количество выполненных итераций.
5. Изменяя значение погрешности, проведите серию вычислений и заполните таблицу 5:

№ испытания	Значение погрешности	Значение корня уравнения	Количество итераций
1	0.1		
2	0.01		
3	0.001		

Сделайте вывод о характере полученных данных.

 © Екатерина Бурова, 2018.

Задания для самостоятельной работы

Задание 5: Существует ряд вычислительных задач, результаты которых приобретают большие значения очень быстро. В связи с этим возникает опасность получить число, выходящее за допустимый диапазон значений используемого типа. К таким задачам, например, можно отнести вычисление факториала. Определите максимальное число, для которого возможно рассчитать факториал, используя для хранения результата типы short, int и long.

Задание 6: Разработайте приложение для вычисления факториала числа. Обеспечьте ввод и проверку исходного числа.

Задание 7: Разработайте приложение для поиска корня уравнения: $5x^2 - 4x - 1 = 0$ на отрезке [1; 2]. Рассчитайте решения при различных значениях приближения.

 © Екатерина Бурова, 2018.