МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра ИС

ОТЧЕТ

по практической работе №1

по дисциплине «Программирование»

Тема: Типы данных и их внутреннее представление в памяти.

Студентка гр. 2373 _____ Андреев Н. С.

Преподаватель _____ Глущенко А. Г.

Санкт-Петербург

2022

Цель работы.

Изучение различных типов данных; применение ввода и вывода типов данных; работа с условной инструкцией, инструкцией множественного выбора и циклами.

Основные теоретические положения.

Тип данных для каждого программного объекта, представляющего данные, определяет: характер данных (число, целое или с дробной частью, одиночный символ или текст и т.д.); объем памяти, который занимают в памяти данные; диапазон возможных значений; правила обработки данных (например, допустимые операции). Типы данных делят на 2 группы: простые и структурированные. Простые типы данных представляют неразделимые данные, не имеющие внутренней структуры (числа, символы и т.д.). Структурированные типы строятся на основе простых типов данных. Другой уровень классификации разделяет все типы данных на предопределенные (встроенные в язык программирования) и пользовательские (определяемые программистом).

Существует 4 спецификатора типа, уточняющих внутреннее представление и диапазон значений стандартных типов: short, long, signed, unsigned.

Переменная — это именованная область памяти, в которой хранятся данные определенного типа. Именем переменной является идентификатор и служит для обращения к области памяти, в которой хранится значение этой переменной. Перед использованием любая переменная должна быть описана.

Программе необходимо точно представлять какие данные хранятся в этом байте памяти.

Для разрешения подобных коллизий в языках программирования введено **понятие типов данных**.

Тип данных для каждого программного объекта, представляющего данные, определяет:

- характер данных (число, со знаком или без знака, целое или с дробной частью, одиночный символ или текст, представляющий последовательность символов и т.д.);
- объем памяти, который занимают в памяти эти данные;
- диапазон или множество возможных значений;
- правила обработки этих данных (например, допустимые операции)

В разных языках программирования определены разные наборы типов данных, но, в целом, типы данных можно разделить на две группы: простые и структурированные типы. Простые типы данных представляют неразделимые данные, не имеющие внутренней структуры (это, например, числа, символы и т.д.). Структурированные типы данных, как это вытекает из их названия, имеют внутреннюю структуру (иногда достаточно сложную). Структурированные типы строятся на основе простых типов данных.

Другой уровень классификации разделяет все типы данных на предопределенные (изначально встроенные в язык программирования) и пользовательские (типы данных, определяемые программистом) типы данных.

Основные (предопределенные) типы данных часто называют арифметическими, поскольку их можно использовать в арифметических операциях.

Типы **int**, **bool** и **char** относят к группе целочисленных (целых) типов, а **float** и **double** - к группе вещественных типов - типов с плавающей точкой. Код, который формирует компилятор для обработки целых величин, отличается от кода для величин с плавающей точкой.

Существует четыре спецификатора типа, уточняющих внутреннее представление и диапазон значений стандартных типов: **short** (короткий); **long** (длинный); **signed** (знаковый); **unsigned** (без знаковый).

Постановка задачи.

Необходимо разработать алгоритм и написать программу, которая позволяет:

1) Вывести, сколько памяти (в байтах) на компьютере пользователя отводится под различные типы данных со спецификаторами и без: int, short int, long int, float, double, long double, char и bool.

2) Вывести на экран двоичное представление в памяти (все разряды) целого числа. При выводе необходимо визуально обозначить знаковый разряд и значащие разряды отступами или цветом.

3) Вывести на экран двоичное представление в памяти (все разряды) типа float. При выводе необходимо визуально обозначить знаковый разряд мантиссы, знаковый разряд порядка (если есть), мантиссу и порядок.

4) Вывести на экран двоичное представление в памяти (все разряды) типа double. При выводе необходимо визуально обозначить знаковый разряд мантиссы, знаковый разряд порядка (если есть), мантиссу и порядок.

Выполнение работы.

Код программы представлен в приложении А.

Описание кода и использованных алгоритмов.

1. При запуске программы появляется окно, в котором выводится количество памяти, которое отводится под различные типы данных на компьютере пользователя. Значения для каждого типа выводятся с новой строки:

```
int 4
short int 2
long int 4
float 4
double 8
long double 8
char 1
bool 1
```

2. Одновременно появляется строка, в которой пользователю предлагается ввести целое число. После ввода строится двоичное представление данного числа, и полученное значение выводится на экран.

```
Введите целое число:  5656
0 0000000 00000000 00010110 00011000
```

3. После вывода двоичного представления целого числа, пользователю предлагается ввести вещественное число, относящееся к типу float. С ним проделывается та же операция. Дробная часть отделяется от целой части точкой.

```
Введите число типа float:  -359.4895
1 - знак  10000111 - экспонента  01100111011111010101000 - мантисса
```

4. Последним вводится значение типа double, и выводится его двоичное представление.

```
Введите число типа double:  15e-48
0 - знак  01101100011 - экспанента  0101111011000010101010010001110011101100110011011011 - мантисса
```

Вывод.

В ходе работы было изучено представление числовых типов данных в памяти компьютера, побитовые операции сдвига и поразрядной конъюнкции, методы перевода целых и вещественных чисел в двоичный код.

ПРИЛОЖЕНИЕ А
КОД ПРОГРАММЫ

```
#include <iostream>
using namespace std;

int main()
{
	setlocale(0, "");

	cout << "int " << sizeof(int) << endl << "short int " << sizeof(short int) << endl << "long int " << sizeof(long int) << endl << "float " << sizeof(float) << endl << "double " << sizeof(double) << endl << "long double " << sizeof(long double) << endl << "char " << sizeof(char) << endl << "bool " << sizeof(bool) << endl << endl;



	int value;
	cout << "Введите целое число" << endl;
	cin >> value;
	unsigned int order = 32;
	unsigned int mask = 1 << order - 1;
	for (int i = 1; i <= order; i++)
	{
		putchar(value & mask ? '1' : '0');
		value <<= 1;
		if (i % 8 == 0)
		{
			cout << " ";
		}
		if (i % order - 1 == 0)
		{
			cout << " ";
		}
	}
	cout << "\n\n";


	union
	{
		float value1;
		int b;
	};
	cout << "Введите число типа float" << endl;
	cin >> value1;
	unsigned int a = sizeof(value1) * 8;
	int mask1 = 1 << a - 1;

	for (int i = 0; i < a; i++)
	{
		if (i == 1)
		{
			cout << " - знак  ";
		}
		if (i % 9 == 0 && i < 10 && i > 2)
		{
			cout << " - экспонента  ";
		}
```

```
		if (b & mask1)
		{
			cout << 1;
		}
		else
		{
			cout << 0;
		}
		b <<= 1;
	}
	cout << " - мантисса" << "\n\n";


	union
	{
		double value2;
		int m[2];
	};
	cout << "Введите число типа double" << endl;
	cin >> value2;
	int mask2 = 1 << (sizeof(int) * 8 - 1);
	for (int i = 0; i < sizeof(int) * 8; i++)
	{
		cout << ((m[1] & mask2) ? 1 : 0);
		if (i == 0)
		{
			cout << " - знак  ";
		}
		if (i > 2 && i < 12 && i % 11 == 0)
		{
			cout << " - экспонента  ";
		}

		m[1] <<= 1;
	}

	for (int i = 0; i < sizeof(int) * 8; i++)
	{
		cout << ((m[0] & mask2) ? 1 : 0);
		m[0] <<= 1;
	}
	cout << " - мантисса" << "\n";

	return 0;
}
```