**Конспект: Обзорная лекция**

Для успешного закрепления пройденного материала по данному учебному модулю, повторим несколько основных понятий:

**Программирование** — это процесс создания компьютерных программ с использованием языка программирования. Оно включает в себя разработку алгоритмов, написание и тестирование программного кода, а также отладку и оптимизацию программ. Программирование позволяет компьютеру выполнять различные задачи, автоматизировать процессы и обрабатывать данные.

**Переменная в программировании** - это именованное место в памяти компьютера, которое используется для хранения и обработки данных.

**Объявление переменной** - это процесс создания переменной в программе путем указания ее имени и типа данных. Объявление переменной сообщает компилятору о существовании переменной определенного типа данных, но не выделяет фактическую память для хранения значения переменной.

**Присваивание значений переменным** - это процесс присвоения конкретного значения переменной. При присваивании значения переменной, вы устанавливаете ее начальное значение или изменяете текущее значение на новое.

**Типы данных в C++** определяют, какие виды значений могут быть хранены и обрабатываться в программе. В C++ существуют различные типы данных, которые предоставляются языком и могут быть использованы для разных целей.

1. Для подготовки к сдаче итогового тестирования за учебный модуль, предлагаем вам вспомнить еще несколько важных моментов из учебных тем:

* Немного определений:

**Транслятор** - программа-переводчик с языка программирования на машинный код. Существует два вида трансляторов: компиляторы и интерпретаторы.

**Компилятор** - это программа-переводчик, которая переводит весь код за один прогон. В результате работы компилятора получаем исполняемый файл.

**Интерпретатор** - это программа переводчик, которая переводит построчно и не создаёт исполняемый файл.

**Важно.** С++ - статически типизированный язык, т.е. мы должны указать тип до использования функции, переменной и т.д. Тип каждой сущности должен быть известен компилятору в точке использования этой сущности.

* Ввод и вывод:

**Вывод в консоль**. Для вывода в консоль мы пользуемся потоком cout. Он принадлежит именному пространству std и прописан в библиотеке <iostream>, без подключения которой работать не будет.

Пример:

#include <iostream> //обязательно нужно

int main() {std::cout << "hello world";}

**Ввод с клавиатуры.** Для ввода с клавиатуры мы пользуемся потоком cin. Он принадлежит именному пространству std и прописан в библиотеке <iostream>.

Пример:

#include <iostream>

int main()

{

int i;

std::cin >> i;

}

* Условный оператор:

**Условный оператор if**.

Структура:

if (условие)

{Нужные действия}

else {действия если условие не выполняется}

Читается так: если условие выполняется (истинно), то выполняем нужные действия. Иначе (если условие ложно) выполняем другие действия.

Оператор **else можно опустить** (убрать), если он не нужен.

В скобках может быть несколько условий. Их объединяют с помощью следующих операторов:

1. **Логическое И: &&**
2. Логическое ИЛИ: ||
3. Логическое НЕ: !

* Циклы:

**Циклы.** Это многократное повторение одного или нескольких действий.

Счётчик цикла - переменная, которая содержит число итераций цикла.

**Цикл -** повторяющиеся действия. **Тело цикла** - последовательность этих действий.

**Итерация** - однократное выполнение тела цикла.

**while**. Цикл с предусловием. Сначала думаем, потом делаем. Сначала проверка условия, потом выполнение.

Структура:

while(условие) {Действия}

Читается так: пока условие истинно, выполняем действия.

**Цикл do-while.** Цикл с постусловием. Сначала делаем, потом думаем. Сначала выполняем первую итерацию цикла, потом проверяем условие.

Структура:

do {действия}

while(условие);

Читается так: выполняем действия, пока условие истинно.

**Важно**. Если мы пишем в условии true, то мы обязаны прописать **break**, иначе программа будет выполнять цикл бесконечно.

while(true)

{break;}

**Цикл for.** Цикл, который проходит по заданному диапазону значений.

Структура:

for(начальное значение счётчика; условие; приращение)

{Действия}

Пример:

for(int i = 0; i < 7; i += 2){}

Повторяем тело цикла, пока i идёт от 0 до 7 с шагом +2 => i = 0, 2, 4, 6 => всего **4 итерации.**

* Массивы:

**Массив.** Это группа данных одного типа, объединённых одним именем.

Элемент массива - переменная с именем массива и индексом.

Индекс элемента показывает его расположение в массиве.

Индексация элементов **начинается с 0**.

**Важно.** В массиве могут быть элементы **только одного типа**.

Массив можно сразу полностью инициализировать, можно и **позже**. Главное - **указать размер** массива.

* Строки:

**Строки** - специальный класс со своими функциями и методами. Позволяет записывать несколько символов в одну переменную. Для использования строк нужно подключить библиотеку <string>.

**Сравнение строк.**

* Строки считаются равными, если все символы обеих строк совпадают. Если хотя бы один символ будет отличаться, то какая-то из строк будет считаться большей, а какая-то - меньшей.
* Когда мы находим первую пару отличающихся символов, мы открываем таблицу ASCII и смотрим, какой из двух символов, которые мы сравниваем, идёт раньше по таблице ASCII. Этот символ будет считаться меньшим => эта строка меньше.
* **Т.е. больше та строка, которая идёт позже в алфавитном порядке**.

**Конкатенация** (объединение). По сути сложение нескольких строк в одну.

* Методы для работы со строками:

**Метод getline().** Используется, чтобы получить предложение (строку с пробелами).

**Метод erase().** Используется для того, чтобы удалить часть строки.

**Метод insert().** Используется для того, чтобы вставить некоторые символы в строку.

**Метод find().** Используется, чтобы найти от какого индекса в строке начинается, нужная последовательность символов. Возвращает целое число, равное индексу.