ENUM

ОСНОВНЫЕ МОМЕНТЫ ИСПОЛЬЗОВАНИЯ ПЕРЕЧИСЛЕНИЙ

1.Использование перечислений вместо констант облегчает чтение программы.

2. Идентификаторы в **enum** должны быть уникальными, но отдельные константы перечисления могут иметь одинаковые значения.

3. Набор идентификаторов перечисляемого типа – собственный уникальный тип, отличающийся от других целочисленных типов.

4. Перечисляемые константы могут определятся и инициализироваься произвольными целочисленными константами, а также константными выражениями:

```
enum ages {Andrey = 47, Ira  <-------  48
      Petya = 56, Yura = Petya + 7};
```

ОСНОВНЫЕ МОМЕНТЫ ИСПОЛЬЗОВАНИЯ ПЕРЕЧИСЛЕНИЙ

5 Каждое перечисление является отдельным типом. Типом элемента перечисления является само перечисление.

тип Keyboard

```
enum Keyword {ASM, AUTO, BREAK };
```

6. Перечисляемая константа может быть обьявлена анонимно, тоесть без имени типа

```
enum {FALSE, TRUE};
```

```
enum {lazy, hazy, crazy} why;
```

Первое обьявление – распространенный способ объявления мнемонический целочисленных констант.

Второе объявление объявляет переменную why с допустимыми значениями lazy, hazy, crazy.

ОСНОВНЫЕ МОМЕНТЫ ИСПОЛЬЗОВАНИЯ ПЕРЕЧИСЛЕНИЙ

7. Перечисления могут неявно преобразовываться в обычные целочисленные типы, но не наоборот

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

void main()
{

    enum boolean { FALSE, TRUE } q;
    enum signal { off, on } a = on; //a инициализируется в on
    enum answer { no, yes, maybe = -1 } b;

    int i, j = true; // верно true преобразуется в 1
    a = off; // верно
    i = a; //верно i становится 0
    q = a; // неверно, два разных типа
    q = (boolean)a // верно, явное преобразование приведением

}
```