Программирование
Тема 8 Перечисления в C++, Java и C#

Перечисления в C++

- **Объявление перечислимого типа:**

```
enum  color {red, green, black}; // вариант 1
const int red = 0;
const int green = 1;
const int black = 2;

enum  color
      { red = 2, green = 2, black = 6 }; // вариант 2
```

- **Объявление переменной перечислимого типа:**

```
enum  color c;    // язык c
color d;          // язык c++
enum  color  {red, green, black} a;

  enum  color c = red;     // правильно
  enum  color c = 0;       // ошибка
  int  i = red;            // правильно
```

Перечисления в C++

- Перечисление - это тип данных, который описывает набор именованных целочисленных констант.
- В примере на слайде enum – ключевое слово, color – имя типа перечисления, r, g, b – сами перечислимые константы.
- При объявлении типа перечисления его значения могут инициализироваться произвольными целочисленными константами или константным выражением.
- Переменным перечислимого типа можно присваивать только именованные значения перечислимых констант.
- Целочисленным переменным можно присваивать значения перечислимых констант.
- Применение перечислений делает программы нагляднее.
- Программа примера 1 посчитывает количество слов в предложении, вводимом с клавиатуры. Слова разделяются одним или несколькими пробелами.

Перечисления в С++

Пример 1. Использование перечислимого типа

```
#include <iostream>
using namespace std;
#include <conio.h>

enum itsaWord { NO, YES };
int main()
{ itsaWord isWord = NO;
  char ch = 'a';
  int wordcount = 0;
  cout << "Enter a phrase:\n";
  do
  { ch = getche();
    if(ch ==' ' || ch =='\r')
     { if( isWord == YES )
        { wordcount++;
          isWord = NO;
        }
     }
    else if( isWord == NO )
              isWord = YES;
 } while( ch != '\r' );   //quit on Enter key
   cout << "\n---Word count is "
        << wordcount << "---\n";
   return 0;
}
```

```
Enter a phrase:
abcd  dfe gfe
---Word count is 3---
```

- **Пример 2.** Создание и использование перечислений

```
package primer;
import java.util.*;

enum Apple {
  Jonathan, GoldenDel, RedDel, Winesap, Cortland
}
class Primer {
  public static void main(String args[])
  {  Apple ap = Apple.RedDel;
    // Output an enum value.
    System.out.println("Value of ap: " + ap);
    System.out.println();
    ap = Apple.GoldenDel;
    // Compare two enum values.
    if(ap == Apple.GoldenDel)
      System.out.println("ap contains GoldenDel.\n");
```

Перечисления в Java

- **Пример 2.** Создание и использование перечислений

```
    // Use an enum to control a switch statement.
    switch(ap) {
      case Jonathan:
        System.out.println("Jonathan is red.");
        break;
      case GoldenDel:
        System.out.println("Golden Delicious is yellow.");
        break;
      case RedDel:
        System.out.println("Red Delicious is red.");
        break;
      case Winesap:
        System.out.println("Winesap is red.");
        break;
      case Cortland:
        System.out.println("Cortland is red.");
        break;
    }
  }
}
```

```
run:
Value of ap: RedDel

ap contains GoldenDel.

Golden Delicious is yellow.
```

Перечисления в Java

- В простейшей форме перечисления в Java подобны перечислениям в C++. Однако это сходство поверхностно. В C++ перечисления представляют совокупность целочисленных констант. В Java перечисления определяют тип класса. За счет реализации перечислений в виде классов сама концепция перечисления значительно расширяется.

- **Пример 3.** Применение методов value() и valueof()

```
package primer;
enum Apple {
  Jonathan, GoldenDel, RedDel, Winesap, Cortland
}
class Primer {
  public static void main(String args[])
  { Apple ap;
    System.out.println("Here are all Apple constants");
    // use values()
    Apple allapples[] = Apple.values();
    for(Apple a : allapples)
      System.out.println(a);
    System.out.println();
    // use valueOf()
    ap = Apple.valueOf("Winesap");
    System.out.println("ap contains " + ap);
  }
}
```

```
run:
Here are all Apple constants
Jonathan
GoldenDel
RedDel
Winesap
Cortland

ap contains Winesap
```

- **Пример 4.** Использование конструктора enum, переменной экземпляра и метода

```
package primer;
enum Apple {
  Jonathan(10), GoldenDel(9), RedDel(12),
                Winesap(15), Cortland(8);
  private int price; // price of each apple
  Apple(int p) { price = p; }  // Constructor
  int getPrice() { return price; }
}
class Primer {
  public static void main(String args[])
  { Apple ap;
    // Display price of Winesap.
    System.out.println("Winesap costs " +
                       Apple.Winesap.getPrice() + " cents.\n");
    // Display all apples and prices.
    System.out.println("All apple prices:");
    for(Apple a : Apple.values())
      System.out.println(a + " costs " +
                         a.getPrice() +  " cents.");
  }
}
```

```
run:
Winesap costs 15 cents.

All apple prices:
Jonathan costs 10 cents.
GoldenDel costs 9 cents.
RedDel costs 12 cents.
Winesap costs 15 cents.
Cortland costs 8 cents.
```

- **Пример 5.** Применение методов ordinal(), compareTo(), and equals().

```
package primer;
enum Apple {
  Jonathan, GoldenDel, RedDel, Winesap, Cortland
}
class Primer {
  public static void main(String args[])
  { Apple ap, ap2, ap3;
    // Obtain all ordinal values using ordinal().
    System.out.println("Here are all apple constants" +
                       " \nand their ordinal values: ");
    for(Apple a : Apple.values())
      System.out.println(a + " " + a.ordinal());
    ap =  Apple.RedDel;
    ap2 = Apple.GoldenDel;
    ap3 = Apple.RedDel;
    System.out.println();
```

- **Пример 5.** Применение методов ordinal(), compareTo(), and equals().

```
    // Demonstrate compareTo() and equals()
    if(ap.compareTo(ap2) < 0)
      System.out.println(ap + " comes before " + ap2);
    if(ap.compareTo(ap2) > 0)
      System.out.println(ap2 + " comes before " + ap);
    if(ap.compareTo(ap3) == 0)
      System.out.println(ap + " equals " + ap3);
    System.out.println();
    if(ap.equals(ap2))
      System.out.println("Error!");
    if(ap.equals(ap3))
      System.out.println(ap + " equals " + ap3);
    if(ap == ap3)
      System.out.println(ap + " == " + ap3);
  }
}
```

```
run:
Here are all apple constants
and their ordinal values:
Jonathan 0
GoldenDel 1
RedDel 2
Winesap 3
Cortland 4

GoldenDel comes before RedDel
RedDel equals RedDel

RedDel equals RedDel
RedDel == RedDel
```

Перечисления в Java

- В

Перечисления в C#

- **Варианты объявления перечислений**

Вариант 1

```
enum days
{
    monday,
    tuesday,
    wednesday,
    thursday,
    friday,
    saturday,
    sunday
}
```

Вариант 2

```
enum time : byte
{
    morning,
    afternoon,
    evening,
    night
}
```

Вариант 3

```
enum operation
{
    add = 2,
    substract = 4,
    multiplay = 8,
    devide = 16
}
```

Вариант 4

```
enum operation
{
    add = 1,    // каждый следующий элемент по умолчанию увеличивается на единицу
    substract, // этот элемент равен 2
    multiplay, // равен 3
    devide     // равен 4
}
```

Перечисления в C#

- Перечисления в C# представляют набор логически связанных констант. Объявление перечисления происходит с помощью оператора enum. Далее идет название перечисления, после которого указывается тип перечисления - он обязательно должен представлять целочисленный тип (byte, int, short, long). Если тип явным образом не указан, то умолчанию используется тип int. Затем идет список элементов перечисления через запятую (см. слайд).
- В примерах на слайде (вариант 1, вариант 2) каждому элементу перечисления присваивается целочисленное значение, причем первый элемент будет иметь значение 0, второй - 1 и так далее. Мы можем также явным образом указать значения элементов, либо указав значение первого элемента (вариант 3, вариант 4).

Перечисления в C#

- **Пример 6.** Использование перечислимого типа

```
using System;
namespace Primer
{ class Program
   { enum Operation
      {  add = 1, substract, multiplay, devide
       }
   static void MathOp(double x, double y, Operation op)
   {  double result = 0.0;
      switch (op)
       {   case Operation.add:
               result = x + y;
               break;
           case Operation.substract:
               result = x - y;
               break;
           case Operation.multiplay:
               result = x * y;
               break;
           case Operation.devide:
               result = x / y;
               break;
       }
       Console.WriteLine("Результат операции равен {0}", result);
   }
```

Перечисления в C#

- **Пример 6.** Использование перечислимого типа

```
        static void Main(string[] args)
        {   // Тип операции задаем с помощью константы
            // Operation.add, которая равна 1
            MathOp(10, 5, Operation.add);
            // Тип операции задаем с помощью константы
            // Operation.multiplay, которая равна 3
            MathOp(11, 5, Operation.devide);
          Console.ReadLine();
        }
    }
}
```

```
Результат операции равен 15
Результат операции равен 2,2
```

Контрольные вопросы

1. Перечислимый тип данных в C++: понятие и примеры использования.
2. Перечислимый тип данных в Java: назначение и примеры использования.
3. Перечислимый тип данных в C#: назначение и примеры использования.