枚举

常量符号化

```c
#include <stdio.h>

const int red = 0;
const int yellow = 1;
const int green = 2;

int main(int argc, char const *argv[])
{
    int color = -1;
    char *colorName = NULL;

    printf("输入你喜欢的颜色的代码: ");
    scanf("%d", &color);
    switch ( color ) {
    case red: colorName = "red"; break;
    case yellow: colorName = "yellow"; break;
    case green: colorName = "green"; break;
    default: colorName= "unknown"; break;
    }
    printf("你喜欢的颜色是%s\n", colorName);

    return 0;
}
```

- 用符号而不是具体的数字来表示程序中的数字

枚举

```c
#include <stdio.h>

enum COLOR {RED, YELLOW, GREEN};

int main(int argc, char const *argv[])
{
    int color = -1;
    char *colorName = NULL;

    printf("输入你喜欢的颜色的代码: ");
    scanf("%d", &color);
    switch ( color ) {
    case RED: colorName = "red"; break;
    case YELLOW: colorName = "yellow"; break;
    case GREEN: colorName = "green"; break;
    default: colorName= "unknown"; break;
    }
    printf("你喜欢的颜色是%s\n", colorName);

    return 0;
}
```

- 用枚举而不是定义独立的const int变量

枚举

- 枚举是一种用户定义的数据类型，它用关键字 enum 以如下语法来声明：

  enum 枚举类型名字 {名字0, …, 名字n} ;

- 枚举类型名字通常并不真的使用，要用的是在大括号里的名字，因为它们就是就是常量符号，它们的类型是int，值则依次从0到n。如：

  enum colors { red, yellow, green } ;

- 就创建了三个常量，red的值是0，yellow是1，而green是2。

- 当需要一些可以排列起来的常量值时，定义枚举的意义就是给了这些常量值名字。

```c
#include <stdio.h>

enum color { red, yellow, green};

void f(enum color c);

int main(void)
{
    enum color t = red;

    scanf("%d", &t);
    f(t);

    return 0;
}

void f(enum color c)
{
    printf("%d\n", c);
}
```

- 枚举量可以作为值
- 枚举类型可以跟上enum作为类型
- 但是实际上是以整数来做内部计算和外部输入输出的

套路：自动计数的枚举

```c
#include <stdio.h>

enum COLOR {RED, YELLOW, GREEN, NumCOLORS};

int main(int argc, char const *argv[])
{
    int color = -1;
    char *ColorNames[NumCOLORS] = {
        "red", "yellow", "green",
    };
    char *colorName = NULL;

    printf("输入你喜欢的颜色的代码: ");
    scanf("%d", &color);
    if ( color >=0 && color < NumCOLORS ) {
        colorName = ColorNames[color];
    } else {
        colorName= "unknown";
    }
    printf("你喜欢的颜色是%s\n", colorName);

    return 0;
}
```

- 这样需要遍历所有的枚举量或者需要建立一个用枚举量做下标的数组的时候就很方便了

枚举量

- 声明枚举量的时候可以指定值
  - enum COLOR { RED=1, YELLOW, GREEN = 5};

```c
#include <stdio.h>

enum COLOR {RED=1, YELLOW, GREEN=5, NumCOLORS};

int main(int argc, char const *argv[])
{
    printf("code for GREEN is %d\n", GREEN);

    return 0;
}
```

枚举只是int

```c
#include <stdio.h>

enum COLOR {RED=1, YELLOW, GREEN=5, NumCOLORS};

int main(int argc, char const *argv[])
{
    enum COLOR color = 0;

    printf("code for GREEN is %d\n", GREEN);
    printf("and color is %d\n", color);

    return 0;
}
```

- 即使给枚举类型的变量赋不存在的整数值也没有任何warning或error

枚举

- 虽然枚举类型可以当作类型使用，但是实际上很(bu)少(hao)用
- 如果有意义上排比的名字，用枚举比const int方便
- 枚举比宏（macro）好，因为枚举有int类型