1、switch...case分支语句

1.1 语法格式

```
switch (表达式)
{
    case 常量表达式1:
        C语句块;
        break;
    case 常量表达式2:
```

```
            C语句块;
            break;
        case 常量表达式3:
            C语言块;
            break;

        ...省略很多case分支...

        default:
            C语句块n;
            break;
}

注:
    1> “表达式”的结果是一个常量，可以是一个普通的变量，
            运算结果为常量的表达式

    2> "常量表达式1" 是一个常量，可以是一个整型的常量，比如1，2，3，4
            可以是一个字符常量，比如:'A', 'B', 'c', 'd'

    3> 每个case分支中都有一个"break;"
break的作用退出switch...case语句，
            break可以省略不写，如果省略break,则继续向下执行case分支语句，
            直到遇到下一个break才会退出switch...case语句。

```

```
    4> default类型与else，是一个托底的，只有以上case分支语句都不满足时，
            才会执行default中对应的C语句块。
            default分支可以省略不写。

执行过程：
    根据“表达式”的结果，直接找到与之匹配的case分支语句，
    然后执行对应的分支语句，遇到break退出switch...case语句，
    如果没有匹配到对应的case分支则会执行default分支中对应的C语句。
```

```
在实际的开发过程中能用switch...case实现的尽量使用switch...case，
尽量不要使用if...else if ... else...。
switch...case的执行效率比if...else的效率要高。
```

1.2 参考案例

```
案例1：将学生成绩划分的代码使用switch...case实现
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int score;
    printf("请输入你的成绩 > ");
    scanf("%d", &score);

```

```c
    if (score < 0 || score > 100)
    {
        printf("输入成绩错误\n");
        return -1;
    }

#if 0
    switch (score / 10)
    {
        case 10:
            puts("等级: A");
            break;
        case 9:
            puts("等级: A");
            break;
        case 8:
            puts("等级: B");
            break;
        case 7:
            puts("等级: C");
            break;
        case 6:
            puts("等级: D");
            break;
        default:
            puts("等级: E");
            break;
    }
#endif
    switch (score / 10)
    {
```

```
40          case 10:
41      /*      puts("等级: A");
42              break;      // 省略break,  继续向下执行知道遇到break, 才会退出
43      */
44          case 9:
45              puts("等级: A");
46              break;
47          case 8:
48              puts("等级: B");
49              break;
50          case 7:
51              puts("等级: C");
52              break;
53          case 6:
54              puts("等级: D");
55              break;
56          default:
57              puts("等级: E");
58              break;
59      }
60
61      return 0;
62  }
63
```

```
1  练习题2: 输入一个字符, 判断字符
2  #include <stdio.h>
3  int main(int argc, const char *argv[])
4  {
5      char ch;
```

```c
    printf("请输入字符a-d > ");
    scanf("%c", &ch);
    switch(ch)
    {
        case 'a':
            printf("ch = %c\n", ch);
            break;
        case 'b':
            printf("ch = %c\n", ch);
            break;
        case 'c':
            printf("ch = %c\n", ch);
            break;
        case 'd':
            printf("ch = %c\n", ch);
            break;
        default:
            printf("其他的字符\n");
            break;
    }


    return 0;
}

```

练习题3：从终端输入年份和月份，判断此月份对应的天数。

```c
#include <stdio.h>

int main(int argc, const char *argv[])
```

```c
{
    int year, mouth;
    printf("请输入年和月份(year mouth) >
");
    scanf("%d%d", &year, &mouth);

    // 在实际编写代码带，注意空格的使用，方便
代码的阅读
    // 运算符两边可以加空格
    if (year <= 0) {
        printf("输入的年份不合理\n");
        return -1;
    }
/*
    if (mouth <= 0 || mouth > 12) {
        printf("输入的月份不合理\n");
        return -2;
    }
*/
    switch(mouth)
    {
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:
            printf("%d年%d月份共计31天
\n", year, mouth);
            break;
```

```
33          case 4:
34          case 6:
35          case 9:
36          case 11:
37              printf("%d年%d月份共计30天
    \n",year, mouth);
38              break;
39          case 2:
40              if (year % 4 == 0 && year %
    100 != 0 || year % 400 == 0) {
41                  printf("%d年是闰年%d月份
    共计29天\n",year,mouth);
42              } else {
43                  printf("%d年是平年%d月份
    共计28天\n",year, mouth);
44              }
45              break;
46          default:
47              printf("输入的月份不合理\n");
48              break;
49      }
50
51
52      return 0;
53  }
54
```

```
1  练习题4: 实现一个两个数的简单的计算器, + - *
   / %
2  请输入一个算数表达式 > 100 + 200
3
```

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int lvalue, rvalue;
    char operator;
    int value;

    printf("请输入一个算数表达式(eg:100 + 200) >");
    scanf("%d%*c%c%*c%d", &lvalue, &operator, &rvalue);

    switch(operator)
    {
    case '+':
        value = lvalue + rvalue;
        break;
    case '-':
        value = lvalue - rvalue;
        break;
    case '*':
        value = lvalue * rvalue;
        break;
    case '/':
        value = lvalue / rvalue;
        break;
    case '%':
        value = lvalue % rvalue;
        break;
    default:
        printf("输入的运算符不合理\n");
```

```
33          break;
34      }
35
36      printf("%d %c %d = %d\n", lvalue, operator, rvalue, value);
37
38
39      return 0;
40  }
41
```

2、while循环

2.1 语法格式

```
1. 语法格式
     while(表达式1)
     {
         C语句块2;
     }

2. 执行过程:
     "表达式1"成立执行"C语句块2", "表达式1"不成立, 退出while循环。

     [1][2,1][2,1][2,1][2,1].......


3. 注意事项:
```

```
14      1> while循环是先判断后执行，C语句块不一定被执行
15      2> 如果C的语句块只有一条C语句，可以省略{},
16      3> 死循环的用法:
17          while(1) { 循环体 }
18          while(1);
19
20
21
```

2.2 练习题

```
1  练习题1：使用while循环实现1+2+3+...100直接的所有的数求和?
2  #include <stdio.h>
3
4  int main(int argc, const char *argv[])
5  {
6      int sum = 0;
7      int i = 1;
8
9      while(i <= 100)  // while (!(i > 100))
10     {
11         sum = sum + i;
12         i++;
13     }
14
15     printf("sum = %d\n", sum);
```

```
16
17      return 0;
18  }
19
```

```
1  练习题2: 求100-999之间的所有的水仙花数?
2      g^3 + s^3 + b^3 = 本身
3
4  #include <stdio.h>
5
6  int main(int argc, const char *argv[])
7  {
8      int i = 100;
9      int g,s,b;
10
11     while(i < 1000)
12     {
13         g = i % 10;
14         s = i / 10 % 10;
15         b = i / 100;
16         if (i == g*g*g + s*s*s + b*b*b)
   {
17             printf("%d ", i);
18         }
19         i++;
20     }
21     puts("");  // 换行符
22
23     return 0;
24 }
25
```

```
练习题3：使用while循环打印以下图形
*
**
***
****
*****
******

#include <stdio.h>
int main(int argc, const char *argv[])
{
    int i, j;
    int line;
    printf("请输入打印三角形的行数 > ");
    scanf("%d", &line);

    i = 0;
    while(i < line)      // 三角形的行数
    {
        j = 0;
        while( j <= i)     // 一列打印几个*
        {
            printf("*");
            j++;
        }
        puts("");
        i++;
    }


```

```
    return 0;
}

```

```
练习题4:
    使用while循环打印九九乘法表,
1 * 1 = 1
1 * 2 = 2   2 * 2 = 4
1 * 3 = 3   2 * 3 = 6   3 * 3 = 9
.........

#include <stdio.h>

int main(int argc, const char *argv[])
{
    int i, j;
    i = 1;
    while(i <= 9)   // 行
    {
        j = 1;
        while(j <= i)   // 列
        {
            printf("%d * %d = %d\t", i, j, i*j);
            j++;
        }
        putchar('\n');
        i++;
    }


```

```
    return 0;
}

```

```
练习题5:
    由1-9，组合一个3位的数，要求各位，十位，百位的数都不相同，
    将所有的可能的情况进行输出。
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int g,s,b;
    g = 1;
    while(g <= 9)
    {
        s = 1;
        while(s <= 9)
        {
            b = 1;
            while(b <= 9)
            {
                if (g != s && s != b && g != b)
                {
                    printf("%d%d%d ", b, s, g);
                    // printf("%d ", b*100+s*10+g);
                }
                ++b;

```

```
24                  }
25                  ++s;
26              }
27              ++g;
28          }
29          putchar('\n');
30          return 0;
31  }
32
33
```

3、do...while循环

```
1.  格式
      do {
          C语句块2;
      }while(表达式);     // 结尾的分号不可以省略

2.  执行过程:
      先执行一次“C语句块2”，在判断“表达式1”是否成立,
      如果成立则继续执行“C语句块2”，如果不成立则退出do...while循环。

      [2,1][2,1][2,1][2,1][2,1].....

3.  注:
      1> do...while循环不管条件表达式是否成立肯定会被执行一次。
```

```
14      2> 如果只有一条语句可以省略{}，一般不要省略。
15      3> do { 循环体; } while(0);    循环体只需要执行1次。
16      4> do...while循环经常和宏定义进行配合使用。
```

```
1  案例1：使用do...while循环打印九九乘法表
2  #include <stdio.h>
3  int main(int argc, const char *argv[])
4  {
5      int i , j;
6      i = 1;
7      do{
8          j = 1;
9          do {
10             printf("%d * %d = %d\t", i,j,i*j);
11         } while(j++ < i);
12         putchar('\n');
13     } while(i++ < 9);
14
15     return 0;
16 }
17
```

```
1  案例：do...while循环和宏定义配合使用的案例。
2  #include <stdio.h>
3
4  // # : 将参数转换为字符串
5  // ## :  字符串的拼接
```

```
 6
 7 #define  PRINT(str,err)
   printf("%s\n",#str);return err
 8
 9 // 宏定义默认要求写到1行，如果分多行进行书写，
   要求加续行符“\”
10 #define  PRI_ERR(str,err) do {
                \
11
   printf("%s\n", #str);  \
12                                  return
   err;              \
13                               } while(0)
14
15 #define PRI_DEBUG(str,err)
   {printf("%s\n", #str); \
16                                     return
   err;          \
17                                  }
18
19 int main(int argc, const char *argv[])
20 {
21     int retValue;
22     retValue = putchar('A');
23     if (retValue == -1)
24     {
25 #if 0
26         printf("put char failed\n");
27         return -1;
28 #endif
```

```
29          // 如果if分支只有1条语句可以省略花括
    号，
30          // 但是这里调用的宏定义，展开之后有多
    条语句，
31          // 因此if的{}不可以省略。
32          PRINT(put char failed, -1);
33      }
34
35      retValue = putchar('B');
36      if (retValue == -1)
37          PRI_ERR(put char failed, -1);
38
39
40      retValue = putchar('C');
41      if (retValue == -1)
42          // PRI_DEBUG(send char failed,
    -1);  // error
43          PRI_DEBUG(send char failed, -1)
      // OK
44      else
45          printf("send char success\n");
46
47      return 0;
48  }
49
```

4、for循环

4.1 for循环的语法格式

```
1. 语法格式
    for(表达式1; 表达式2; 表达式3)    // 常见
    {
        C语句块4;
    }

    表达式1：一般为赋值语句，比如i = 1
    表达式2：循环结束的判断语句，结果为真或假，比如1 <= 9
    表达式3：改变循环的语句，比如i++  j++
    C语句块4：for循环的循环体

2. 执行过程
    先执行表达式1，在执行表达式2，如果表达式2成立，则执行C语句块4，
    如果不成立则退出for循环；执行完C语句块4之后，执行表达式3，
    然后再执行表达式2，如果表达式2成立，再执行C语句块4，
    如果不成立则退出for循环；依次类推。

    [1,2][4,3,2][4,3,2][4,3,2][4,3,2].....

3. for循环的表达式可以省略
    for(;;)      // 常见
    {
        循环体;
    }   ----> 死循环

```

```
26      for(;表达式2;表达式3)     // 常见
27      {
28          循环体;
29      }    ----> 要求再for循环之前进行赋初值
30      for(表达式1;;表达式3)      // 不常见
31      {
32          循环体;
33      } ----> 死循环
34
35      for(表达式1;表达式2;)      // 不常见
36      {
37          循环体;
38      }
```

4.2 参考案例

```
1 前边使用while循环完成的案例，都可以使用for循环实现。（自己课下完成）
```

```
1 案例1：  将一个无符号的整数，使用二进制的形式进行打印。
2       unsigned int num = 0x12345678;
3
4 #include <stdio.h>
5
6 int main(int argc, const char *argv[])
7 {
8       unsigned int num =0x12345678;
9
10      int i;
```

```
    printf("0b");
    for (i = 0; i < 32; i++)
    {

        if (num & (0x1 << (31 - i))) //
num最高位与1，其他为与0
        {
            printf("%d", 1);
        }
        else
        {
            printf("%d", 0);
        }
        if ((i + 1) % 4 == 0)
        {
            printf(" ");
        }
        // num <<= 1;
    }

    printf("\n");
    return 0;
}

```

```
案例2：for循环嵌套的
    *
   **
  ***
 ****
*****
```

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int i,j;
    int line;
    printf("请输入打印的行数 > " );
    scanf("%d", &line);
    for(i = 1; i <= line; i++)  // 打印行数
    {
        for(j = 1; j <= line - i; j++)  // 打印一行有几个空格
        {
            printf(" ");
        }
        for(j = 1; j <= i; j++)  // 打印一行有几个*
        {
            printf("*");
        }
        printf("\n");

    }


    return 0;
}

```

案例3：从终端输入一个大于0整数，求出阶乘的之和结果。

```
    比如输入5,  1! + 2! + 3! + 4! + 5! =
                  1 + 1*2 + 1*2*3 + 1*2*3*4 + 1*2*3*4*5 =
    外层循环求和;
    内存循环求阶乘;
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int i,j, num;
    long mul_val = 1;
    long sum_val = 0;

    printf("请输入一个整数 > ");
    scanf("%d", &num);
    for (i = 1; i <= num; i++)
    {
        for (mul_val = 1, j = 1; j <= i; j++)
        {
            mul_val *= j;
        }
        sum_val += mul_val;
    }
    printf("%ld\n", sum_val);

    return 0;
}
```

案例4：
求出用50元， 20元， 10元， 可以组合成100元的所有的情况?

```
50元      20元      10元
0         0         0
0         0         1
0         0         2
0         0         3
......
0         1         0
0         1         1
0         1         2
0         1         3
.....
```

```c
#include <stdio.h>
int main(int argc, const char *argv[])
{
    int i,j,k;
    int sum = 0;

    for (i = 0; i <= 2; i++)
    {
        for(j = 0; j <= 5; j++)
        {
            for(k = 0; k <= 10; k++)
            {
                if (i*50+j*20 + k*10 == 100)
                    printf("50元%d张, 20元%d张, 10元%d张\n", i,j,k);
```

```
            }
        }
    }


    return 0;
}

```

```
案例5：使用for循环嵌套，打印以下图形。
    FEDCBA
    +FEDCB
    ++FEDC
    +++FED
    ++++FE
    +++++F

#include <stdio.h>
int main(int argc, const char *argv[])
{
    int i,j;

    for(i = 0; i < 6; i++)       // 打印行数
    {
        for(j = 0; j < i; j++)  // 打印+字符，一列打印几个+
        {
            putchar('+');
        }

```

```
21          for(j = 0; j < 6 - i; j++)  //
   打印F-A字母
22          {
23              putchar('F'-j);
24          }
25          putchar('\n');
26      }
27
28
29      return 0;
30  }
31
32
```

```
1  明天的授课内容:
2      break continue return
3      函数
4      数组
```