程序设计习题课

2021-09-30

内容

1. 复习回顾
2. 小测试

复习回顾

数据类型

1. 基本数据类型
    1. 整型：　short, int, long;
        - 溢出
            - 上溢出
            - 下溢出
        - 有符号数/无符号数
            - 区别：主要在于二进制表示的最高位是否用于表示符号
            - 两者表示的区间长度一致，但区间范围有所偏移
    2. 浮点型：　float, double, long double;
    3. 字符型：　char;
        - 常用转义字符：\n \t \v \r \\ \' \" \0 \ddd \xhh
        - **单字符变量使用''进行赋值，字符串变量使用""进行赋值**，字符串本质上是字符数组。
        - 字符型数据与整型数据通用。
2. 指针类型
    - 内存地址
3. 结构化数据类型
    - 其他基本类型的结合，如数组、结构、联合、枚举等。

常量

- 值无法改变的数据，可以是任意数据类型。
- 一般使用宏定义预处理命令对常量进行定义，同时常量名一般为全大写的形式。

```
#define PI 3.1415926
#define MAXEPOCHS 300
```

变量

- 与常量相对，是值可以改变的数据，支持任意数据类型。
- 变量定义时需要指明相应的数据类型，同时可以为其赋初值。

```
int idx = 1;
float inp = 1.5e3;
char * word;
```

命名规范

1. 只能由字母、数字、下划线组成；
2. 第一个字符必须是英文字母或下划线；
3. 有效长度为255个字符；
4. 不可以包含标点符号和类型说明符（%、&、！、#、@、$）；
5. 不可以是关键字；
    - 命名一般要体现**该变量或该函数的作用**，尽量避免使用过于简单的变量名，如a,s,x等，方便以后修改或回顾代码时快速理解。
    - 对于多个单词组合的情形，有两类命名方式：
        1. 驼峰命名法：minAreaRect, GetWindowRect, SetCursorPos;
        2. 下划线命名法：min_area_rect, get_window_rect, set_cursor_pos;

- **关键字**
    - 【存储类型】**auto, extern, register, static;**
    - 【程序控制】**break, case, continue, default, do, else, for, goto, if, return, switch, while;**
    - 【数据类型】**char, double, enum, float, int, long, short, signed, struct, union, unsigned, void;**
    - 【其他】**const, volatile, typedef, sizeof;**
    - 【预处理】**define, undef, include, ifdef, ifndef, endifline, elif, line;**

输入输出

1. 单字符输出和输入
    1. **int putchar(int)**
        - 该函数以无符号 char 强制转换为 int 的形式返回写入的字符。
        - 当输出正确的时候，返回输出字符转换为的unsigned int 值；
        - 当输出错误的时候，返回 EOF（End of file）文件结束符。
    2. **int getchar(void)**
        - 该函数以无符号 char 强制转换为 int 的形式返回读取的字符，如果到达文件末尾或发生读错误，则返回 EOF。
2. 格式输出和输入
    1. **int printf(const char *, ...)**
    - 函数返回值为整型。若成功则返回输出的字符数，输出出错则返回负值。

格式符	含义
d或i	十进制输出整数
o	八进制输出整数
x或X	十六进制输出整数
u	无符号十进制输出整数
F	小数输出单、双精度浮点数
e或E	指数输出单、双精度浮点数
c	输出单字符
s	输出字符串
%	输出%

修饰符	含义
-	左对齐
+	正数输出带正号
#	八进制输出带0，十六进制输出带0x，浮点数输出带.
数字	**指定数据输出的宽度，*表示当前输出宽度由下一输出宽度指明**
.数字	**对于浮点数，指明小数点之后显示位数；对于字符串，指明输出字符数**
H	输出短整型数
l或L	输出long int或long double

- 注意事项
  - 域宽w
    - 若数据长度大于w，输出实际数据
    - 若数据长度小于w，补充空格（前导0则补充0）
    - 若为*，则由参数列表指定
  - 精度.p
    - 浮点数：指定小数点后位数，默认6位
    - 字符串：指定输出的字符数
    - 整数：输出最小位数，不足补0

2. **int scanf(const char *, ...)**

- 返回成功读入的数据项数，读入数据时遇到了“文件结束”则返回EOF。

格式符	含义
d或i	十进制输入整数
o	八进制输入整数
x或X	十六进制输出整数
u	无符号十进制输入整数
f或e	输入浮点数
c	输入 单字符
s	输入字符串

修饰符	含义
*	**赋值抑制符，不将当前输入赋值给变量**
数字	指定输入数据的数字位数
H	输入short整数
l或L	输入long整数或long double浮点数

3. 例题

```
printf("%#o,%4o,%6lo\n",045,045,-1)  // 045,  45,37777777777
printf("%d,%4u,%lu",4294967295u,4294967295u,-1)
//-1,4294967295,4294967295
```

程序结构

1. 顺序结构
2. 选择结构
    - if ... else ...
    - switch ... case ...
3. 循环结构
    - for ...
    - while ...
    - do ... while ...

数制

进制转换

- 二进制、八进制、十六进制、十进制相互转换

表示法

1. 原码：最高位表示符号，其余位为二进制表示
2. 反码：正数为原码本身；负数除最高位不动，其余位对原码取反
3. 补码：补码为原码本身；负数在反码基础上+1

小测试

指出以下不符合命名规范的命名方式

1. static
2. _age
3. union
4. __version
5. conv1*1
6. 1stplace
7. validInput
8. plot_contours
9. special!number!
10. case
11. end
12. tmp-var

写出 $12_{(10)}$ 的二进制和十六进制表示

写出 $12_{(10)}$和$-2_{(10)}$二进制下的原码、反码和补码（假设8位数据）

编写程序，交换两个整数的值（使用额外的中间变量，不使用额外变量）

1. 1,3,5,6,9,10,12;
2. 1100, C;
3. 00001100, 00001100, 00001100;
   10000010, 11111101, 11111110
4. 参考答案

```c
#include <stdio.h>

int main()
{
    int x = 1, y = 2;
    int tmp;

    printf("%d %d\n", x, y);
    // 使用中间变量
    tmp = x;
    x = y;
    y = tmp;
    printf("%d %d\n", x, y);

    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    int x = 1, y = 2;

    printf("%d %d\n", x, y);
    // 不使用中间变量
    x = x + y;
    y = x - y;
    x = x - y;
    printf("%d %d\n", x, y);

    return 0;
}
```

```c
#include <stdio.h>

int main()
{
    int x = 1, y = 2;

    printf("%d %d\n", x, y);
    // 不使用中间变量
    x = x ^ y;
    y = x ^ y;
    x = x ^ y;
    printf("%d %d\n", x, y);

    return 0;
}
```