

# 2025.11.11 C语言程序设计上机实习四

## 必做题

NWAFU-OJ  主页  竞赛&作业列表  问题列表  状态列表  排名  OH-排名  2025013665(马建宇)				
信息与公告				
状态	题目编号	标题	正确	提交量
Y	A	C语言实习一--1.简单程序设计	473	715
Y	B	C语言实习一--2.简单计算	472	1348
Y	C	C语言实习一--3.计算梯形面积	456	778
Y	D	C语言实习一--4.输出字符对应的ASCII码	443	557
Y	E	C语言实习一--5.数据类型长度测试	436	845
Y	F	C语言实习二--1.成绩判断	438	791
Y	G	C语言实习二--2.字符转换	431	1006
Y	H	C语言实习二--3.利用海伦公式求三角形面积	417	1773
Y	I	C语言实习二--4.判断是否能构成一个三角形	406	593
Y	J	C语言实习二--5.按从大到小排序三个数	404	1880
Y	K	C语言实习三--1.数据统计	374	948
Y	L	C语言实习三--2.斐波那契阶梯问题	379	640
Y	M	C语言实习三--3.猴子吃桃问题	358	670
Y	N	C语言实习三--4.求两个数的最大公约数和最小公倍数	330	940
Y	O	C语言实习三--5.图形输出	317	888
Y	P	C语言实习四--1.字符串连接	190	371
Y	Q	C语言实习四--2.简单加密程序	167	442

## 选做题

题目源文件[实验四.pdf](#)

完整源代码

```
#include <stdio.h>

int Char2Int(char ch); // 将字符 '0'-'9','A'-'F' 转换为对应的整数值 (0~15)
int Strlen(char *str); // 计算字符串长度 (不使用 string.h)
char Int2Char(int d); // 将 0~15 转为 '0'-'9' 或 'A'-'F'
int Power(int base, int exp); // 计算 base^exp (exp ≥ 0)
int Base2Dec(int base, char *num); // 将 base 进制字符串转为十进制整数
void Dec2Base(int num, int base, char *converted); // 十进制转 base 进制 (低位在前)
void Reverse(char *str); // 反转字符串 (原地操作)
void Base2Base(int base_from, int base_to, char *num_from, char *num_to);

int main(void)
{
    int base_from, base_to; // base_from为原始进制, base_to为目标进制
    char num_from[11]; // 输入原始字符串, 最大支持10字符, 留1给'\0'
    char num_to[11]; // 输出转换后字符串, 最大支持10字符, 留1给'\0'
    scanf("%d %s %d", &base_from, num_from, &base_to); // scanf格式化输入"原始进制 原始字符串 目标进制"
    Base2Base(base_from, base_to, num_from, num_to); // 调用'Base2Base进制', 传入四个参量:"原始进制,目标进制,原始字符串,转换后字符串"
    printf("%s\n", num_to); // 格式化输出转换后字符串
    return 0;
}
```

```
// 将字符 '0'-'9','A'-'F' 转换为对应的整数值 (0~15)
// 若字符非法 (如 'G', 'z'), 返回 -1
// 调用返回整型数字
```

```
int Char2Int(char ch)
{
    if (ch >= '0' && ch <= '9')
        return ch - '0'; // 将字符转化为数字
    switch (ch) // 大写字母的转换
    {
        case 'A': return 10;
        case 'B': return 11;
        case 'C': return 12;
        case 'D': return 13;
        case 'E': return 14;
        case 'F': return 15;
        default: return -1; // 字符非法处理
    }
}
```

```
// 计算字符串长度 (不使用 string.h)
// 调用返回传入字符串数组的长度, 不包含'\0'
```

```
int Strlen(char *str)
{
    int i = 0;
    while (str[i] != '\0') // 使用while来遍历字符, 直至到字符串终止符'\0'
        i++;
    return i; // 返回字符串长度
}
```

```
// 将字符 '0'-'9','A'-'F' 转换为对应的整数值 (0~15)
// 若整数非法 (如 'G', 'z'), 返回 -1
// 调用返回传入整数的对应字符
```

```
char Int2Char(int d)
{
    switch (d) //传入整数的转换
    {
        case 0: return '0';
        case 1: return '1';
        case 2: return '2';
        case 3: return '3';
        case 4: return '4';
        case 5: return '5';
        case 6: return '6';
        case 7: return '7';
        case 8: return '8';
        case 9: return '9';
        case 10: return 'A';
        case 11: return 'B';
        case 12: return 'C';
        case 13: return 'D';
    }
}
```

```

        case 14: return 'E';
        case 15: return 'F';
        default: return -1; // 整数非法处理
    }
}

// 计算 base^exp(exp ≥ 0)
// 若exp<0, 返回-1
// 调用返回base^exp的结果
int Power(int base, int exp)
{
    int result = 1;
    if ( exp >= 0 )
    {
        for ( int i = 0; i < exp; i++ )
        {
            result *= base;
        }
        return result;
    }
    else
    {
        printf("幂不能为负数!");
        return -1;
    }
}

// 将 base 进制字符串转为十进制整数
// 调用返回字符串对应的十进制整数
int Base2Dec(int base, char *num)
{
    int len = strlen(num);
    int result;

    if ( base < 2 || base > 16 )
    {
        printf("本程序仅支持 2 到 16 进制 之间的相互转换!请检查输入!");
        return -1;
    }

    for ( int i = 0; i < len; i++ )
    {
        int vaild = Char2Int(num[i]);
        if ( vaild < 0 || vaild >= base )
        {
            printf("输入的字符串与初始进制不对应!");
            return -1;
        }
        result += vaild * Power(base, len - i - 1);
    }
    return result;
}

```

```
}
```

```
// 十进制转 base 进制（低位在前）
```

```
// 调用后输出低位在前的converted数组
```

```
void Dec2Base(int num, int base, char *converted)
```

```
{
```

```
    int i = 0;
```

```
    int temp;
```

```
    if ( base < 2 || base > 16 )
```

```
    {
```

```
        printf("本程序仅支持 2 到 16 进制 之间的相互转换!请检查输入!");
```

```
        converted[0] = '\0';
```

```
        return;
```

```
    }
```

```
    if ( num == 0 )
```

```
    {
```

```
        converted[0] = '0';
```

```
        converted[1] = '\0';
```

```
        return;
```

```
    }
```

```
    while ( num > 0 )
```

```
    {
```

```
        temp = num % base;
```

```
        converted[i++] = Int2Char(temp);
```

```
        num /= base;
```

```
    }
```

```
    converted[i] = '\0';
```

```
    return;
```

```
}
```

```
// 反转字符串（原地操作）
```

```
// 使用按位异或，不会开辟新的内存空间
```

```
void Reverse(char *str)
```

```
{
```

```
    int len = Strlen(str);
```

```
    int i = 0, j = len - 1;
```

```
    while ( i < j )
```

```
    {
```

```
        str[i] ^= str[j];
```

```
        str[j] ^= str[i];
```

```
        str[i] ^= str[j];
```

```
        i++;
```

```
        j--;
```

```
    }
```

```

    return;
}

// 主要函数
void Base2Base(int base_from, int base_to, char *num_from, char *num_to)
{
    int num_Dec; // 存储原始字符串转换后的十进制数

    if ( base_from < 2 || base_from > 16 || base_to < 2 || base_to > 16 )
    {
        printf("本程序仅支持 2 到 16 进制 之间的相互转换!请检查输入!");
        return;
    }

    num_Dec = Base2Dec(base_from, num_from); // 赋值, 调用Base2Dec函数

    if ( num_Dec < 0 )
    {
        num_to[0] = '\0';
        return;
    }

    Dec2Base(num_Dec, base_to, num_to); // 使用Dec2Base将十进制数转换为目标进制的字符串(低位在前)

    Reverse(num_to); // 反转(高位在前)

    return;
}

```

## 编译&&测试

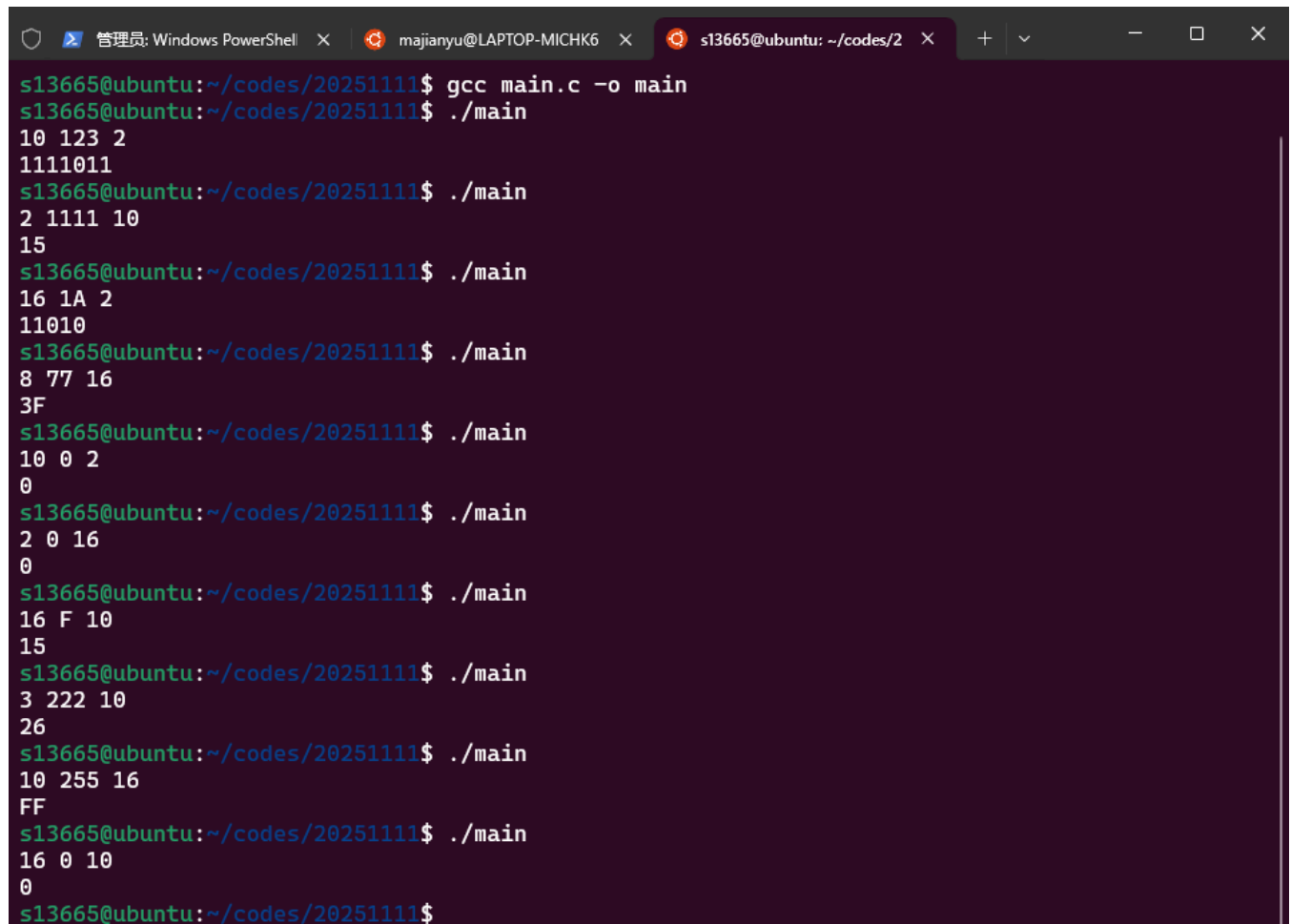
```

s13665@ubuntu:~/codes/20251111$ gcc main.c -o main
s13665@ubuntu:~/codes/20251111$ ./main
10 123 2
1111011
s13665@ubuntu:~/codes/20251111$ ./main
2 1111 10
15
s13665@ubuntu:~/codes/20251111$ ./main
16 1A 2
11010
s13665@ubuntu:~/codes/20251111$ ./main
8 77 16
3F
s13665@ubuntu:~/codes/20251111$ ./main
10 0 2
0
s13665@ubuntu:~/codes/20251111$ ./main
2 0 16
0
s13665@ubuntu:~/codes/20251111$ ./main
16 F 10

```

```
15
s13665@ubuntu:~/codes/20251111$ ./main
3 222 10
26
s13665@ubuntu:~/codes/20251111$ ./main
10 255 16
FF
s13665@ubuntu:~/codes/20251111$ ./main
16 0 10
0
s13665@ubuntu:~/codes/20251111$
```

截图



```
s13665@ubuntu:~/codes/20251111$ gcc main.c -o main
s13665@ubuntu:~/codes/20251111$ ./main
10 123 2
1111011
s13665@ubuntu:~/codes/20251111$ ./main
2 1111 10
15
s13665@ubuntu:~/codes/20251111$ ./main
16 1A 2
11010
s13665@ubuntu:~/codes/20251111$ ./main
8 77 16
3F
s13665@ubuntu:~/codes/20251111$ ./main
10 0 2
0
s13665@ubuntu:~/codes/20251111$ ./main
2 0 16
0
s13665@ubuntu:~/codes/20251111$ ./main
16 F 10
15
s13665@ubuntu:~/codes/20251111$ ./main
3 222 10
26
s13665@ubuntu:~/codes/20251111$ ./main
10 255 16
FF
s13665@ubuntu:~/codes/20251111$ ./main
16 0 10
0
s13665@ubuntu:~/codes/20251111$
```