1. 实数的整数次幂

REAL Power(REAL x, INTEGER n);

求任意实数的任意整数次幂。用下面的主函数进行测试。

```
int main()
{
    REAL x, y;
    INTEGER n;
    printf("底数: ");
    scanf(RSF, &x);
    printf("指数: ");
    scanf(ISF, &n);
    y = Power(x, n);
    printf("结果: " RPF "\n", y);
    return 0;
}
```

测试用例:

底数: 0.000000001↵ 指数: 5↵ 结果: 0	底数: 0.000000001↵ 指数: 0↵ 不正确的参数!	底数: 0.000000001↵ 指数: -2↵ 不正确的参数!	底数: 2.5↵ 指数: 0↵ 结果: 1
底数: 2.5↵ 指数: 3↵ 结果: 15.625	底数: 2.5↵ 指数: -3↵ 结果: 0.064	底数: 1.000001↵ 指数: 1000000↵ 结果: 2.71828046909594	底数: 1 ↵ 指数: 10000000000↵ 结果: 1

提示:

当$x = 0$时, 若$n > 0$, 则$x^n = 0$;

若$n \le 0$, 则无意义。

当$x \ne 0$时, 若$n = 0$, 则$x^n = 1$;

若$n > 0$, 则 $x^n = \underbrace{x \cdot x \cdot \cdots \cdot x}_{n个}$ (如 $x^3 = \underbrace{x \cdot x \cdot x}_{3个}$);

若$n < 0$, 则 $x^n = \dfrac{1}{\underbrace{x \cdot x \cdot \cdots \cdot x}_{-n个}}$ (如 $x^{-3} = \dfrac{1}{\underbrace{x \cdot x \cdot x}_{3个}}$)。

2. 多项式求和

$$f(x, n) = 1 - x + x^2 - x^3 + \ldots + (-1)^{n-1}x^n$$

REAL F(REAL x, INTEGER n);

用下面的主函数进行测试。

```
int main()
{
    REAL x, y;
    INTEGER n;
    printf("x = ? ");
    scanf(RSF, &x);
    printf("n = ? ");
    scanf(ISF, &n);
    y = F(x, n);
    printf("结果: " RPF "\n", y);
    return 0;
}
```

测试用例:

x = ? 0.5↵ n = ? 0↵ y = 1	x = ? 0.5↵ n = ? 1↵ y = 0.5	x = ? 0.5↵ n = ? 2↵ y = 0.75	x = ? 0.5↵ n = ? 3↵ y = 0.625	x = ? 0.5↵ n = ? 25↵ y = 0. 666666656732559	x = ? 0.5↵ n = ? 49↵ y = 0. 666666666666666

3. 正整数的最高位与最低位数字

INTEGER TopDigit(INTEGER x);

INTEGER LowDigit(INTEGER x);

说明: 1048576的最高位是1, 最低位是6。用下面的主函数来进行测试。

```
int main()
{
    INTEGER x, a, b;
    printf("正整数: ");
    scanf(ISF, &x);
    a = TopDigit(x);
    b = LowDigit(x);
    printf("最高位: " IPF "\n", a);
    printf("最低位: " IPF "\n", b);
    return 0;
}
```

测试用例:

正整数: 32768↵ 最高位: 3 最低位: 8	正整数: 4185371293↵ 最高位: 4 最低位: 3	正整数: 9223372036854775807↵ 最高位: 9 最低位: 7

提示: 将整数反复除以10到只剩一位数字时即得到最高位。

4. 正整数的第*n*位数字

INTEGER Digit(INTEGER x, INTEGER n);

说明: 整数的数位由低到高从0开始编号, 如: 1048576第0位(即个位)数字是6, 第1位(即十位)数字是7, 第2位(即百位)数字是5, …, 第5位(即十万位)数字是0, 第6位(即百万位)数字是1, 第7位及以上的数字是0。

…	9	8	7	6	5	4	3	2	1	0
…	0	0	0	1	0	4	8	5	7	6

用下面的主函数来进行测试。

```
int main()
{
    INTEGER x, n, y;
    printf("正整数: ");
    scanf(ISF, &x);
    printf("位　号: ");
    scanf(ISF, &n);
    y = Digit(x, n);
    printf("数　字: " IPF "\n", y);
    return 0;
}
```

测试用例:

正整数: 32768↵ 位　号: 3 数　字: 2	正整数: 4185371293↵ 位　号: 0 数　字: 3	正整数: 9223372036854775807↵ 位　号: 19 数　字: 0

5. 正整数的位数

INTEGER NumDigit(INTEGER x);

说明: 1048576的位数为7。用下面的主函数来进行测试。

```
int main()
{
    INTEGER x, n;
    printf("正整数: ");
    scanf(ISF, &x);
    n = NumDigit(x);
    printf("位　数: " IPF "\n", n);
    return 0;
}
```

测试用例:

正整数: 32768↵ 位　数: 5	正整数: 4185371293↵ 位　号: 10	正整数: 9223372036854775807↵ 位　号: 19

6. 整数各位数字之和

INTEGER SumDigit(INTEGER x);

说明: 1048576各位数字之和为1 + 0 + 4 + 8 + 5 + 7 + 6 = 31。用下面的主函数来进行测试。

```
int main()
{
    INTEGER x, y;
    printf("正整数: ");
    scanf(ISF, &x);
    y = SumDigit(x);
    printf("各位数字之和: " IPF "\n", y);
    return 0;
}
```

测试用例:

正整数: 32768↵ 各位数字之和: 26	正整数: 4185371293↵ 各位数字之和: 43	正整数: 9223372036854775807↵ 各位数字之和: 88

7. 展品陈列

有6件展品分给3个展台陈列, 要求: 一个展台1件、一个展台2件、一个展台3件。请问一共有多少种分法?

1) 阶乘函数

REAL Fac(INTEGER n);

函数值为自然数n的阶乘。用下面的主函数来进行测试。

```
int main()
{
    INTEGER n;
    REAL f;
    printf("n = ? ");
    scanf(ISF, &n);
    f = Fac(n);
    printf("f = " RPF "\n", f);
    return 0;
}
```

测试用例:

n = ? 0↵ f = 1	n = ? 4↵ f = 24	n = ? 15↵ f = 1307674368000	n = ? 70↵ f = 1.19785716699699e+100	n = ? -1↵ 不正确的参数!

2) 排列函数

REAL Arg(INTEGER m, INTEGER n);

要求: 若参数不正确, 则报告错误, 强行结束程序。为与阶乘函数、组合函数保持兼容, 允许$n = 0$, 并规定函数值为1。用下面的主函数来进行测试。

```
int main()
{
    INTEGER m, n;
    REAL a;
    printf("m = ? ");
    scanf(ISF, &m);
    printf("n = ? ");
    scanf(ISF, &n);
    a = Arg(m, n);
    printf("a = " RPF "\n", a);
    return 0;
}
```

测试用例:

m = ? 10↵ n = ? 4↵ a = 5040	m = ? 10↵ n = ? 0↵ a = 1	m = ? 50↵ n = ? 25↵ a = 1.96078146816082e+039	m = ? 3↵ n = ? 4↵ 不正确的参数!	m = ? 5↵ n = ? -1↵ 不正确的参数!

3) 组合函数

REAL Cmb(INTEGER m, INTEGER n);

要求: 若参数不正确, 则报告错误, 强行结束程序。

用下面的主函数来进行测试。

```
int main()
{
    INTEGER m, n;
    REAL c;
    printf("m = ? ");
    scanf(ISF, &m);
    printf("n = ? ");
    scanf(ISF, &n);
    c = Cmb(m, n);
    printf("c = " RPF "\n", c);
    return 0;
}
```

测试用例:

m = ? 10↵ n = ? 4↵ c = 210	m = ? 10↵ n = ? 0↵ c = 1	m = ? 34↵ n = ? 17↵ a = 2333606220	m = ? 3↵ n = ? 4↵ 不正确的参数!	m = ? 5↵ n = ? -1↵ 不正确的参数!

4) 在下面的主函数中填写适当内容, 计算并输出展品分配给展台的分法总数。(答案: 360种)

```
int main()
{
    printf( RPF, _____ );
    return 0;
}
```

提示: 这里计算的是分法, 不是摆法。因此展品分配给展台用组合, 但先分配给哪一个展台、后分配给哪一个展台却有先后次序, 因此这里需要用到排列。

8. *爬楼梯

*n*级台阶的楼梯, 每一步可以跨1到3级台阶, 共有多少种爬楼方案?

REAL Upstairs(INTEGER n);

用下面的函数进行测试。

```
int main()
{
    INTEGER n;
    REAL u;
    printf("台阶 = ? ");
    scanf(ISF, &n);
    u = Upstairs(n);
    printf("方案 = " RPF "\n", u);
    return 0;
}
```

测试用例:

台阶 = ? 1↵ 方案 = 1	台阶 = ? 2↵ 方案 = 2	台阶 = ? 3↵ 方案 = 4	台阶 = ? 4↵ 方案 = 7	台阶 = ? 10↵ 方案 = 274	台阶 = ? 39↵ 方案 = 12960201916

9. 任意数制输入任意整数

1) 判断数字

BOOL IsBinDigit(char x);

若x为二进制数数字'0'或'1', 则函数值为1(真), 否则为0(假)。

BOOL IsOctDigit(char x);

若x为八进制数数字'0' ~ '7', 则函数值为1(真), 否则为0(假)。

BOOL IsDecDigit(char x);

若x为十进制数数字'0' ~ '9', 则函数值为1(真), 否则为0(假)。(注: 虽然此函数与系统库函数isdigit功能相当, 但作为练习请自编此函数完成判断功能。)

BOOL IsHexDigit(char x);

若x为十六进制数数字'0' ~ '9'、'a' ~ 'f'、'A' ~ 'F', 则函数值为1(真), 否则为0(假)。(注: 虽然此函数与

系统库函数isxdigit功能相当，但作为练习请自编此函数完成判断功能。)

用下面的主函数来进行测试。

```
int main()
{
    char x;
    printf("字符: ");
    scanf("␣%c", &x);
    if (IsBinDigit(x))
    {
        puts("二进制数字");
    }
    if (IsOctDigit(x))
    {
        puts("八进制数字");
    }
    if (IsDecDigit(x))
    {
        puts("十进制数字");
    }
    if (IsHexDigit(x))
    {
        puts("十六进制数字");
    }
    return 0;
}
```

测试用例:

字符: 1↵ 二进制数字 八进制数字 十进制数字 十六进制数字	字符: 5↵ 八进制数字 十进制数字 十六进制数字	字符: 8↵ 十进制数字 十六进制数字	字符: f↵ 十六进制数字	字符: K↵

2) 输入整数

BOOL ScanBin(INTEGER *x);

以二进制输入自然数，保存到指针x所指示的变量中。

BOOL ScanOct(INTEGER *x);

以八进制输入自然数，保存到指针x所指示的变量中。

BOOL ScanDec(INTEGER *x);

以十进制输入自然数，保存到指针x所指示的变量中。

BOOL ScanHex(INTEGER *x);

以十六进制输入自然数，保存到指针x所指示的变量中。

用下面的主函数来进行测试。

```
int main()
{
    INTEGER r, x;
    BOOL ok = TRUE;
    printf("基数: ");
    scanf(ISF, &r);
    switch (r)
    {
    case 2:
        printf("二进制: ");
        ScanBin(&x);
        break;
    case 8:
        printf("八进制: ");
        ScanOct(&x);
        break;
    case 10:
        printf("十进制: ");
```

```
        ScanDec(&x);
        break;
    case 16:
        printf("十六进制: ");
        ScanHex(&x);
        break;
    default:
        puts("不正确的数制!");
        ok = FALSE;
    }
    if (ok)
    {
        printf("十进制: " IPF "\n", x);
    }
    return 0;
}
```

测试用例:

基数: 2↵ 二进制: 110110010011↵ 十进制: 3475	基数: 8↵ 八进制: 6623↵ 十进制: 3475	基数: 10↵ 十进制: 3475↵ 十进制: 3475	基数: 16↵ 十六进制: d93↵ 十进制: 3475
基数: 2↵ 二进制: 1101100100112↵ 十进制: 3475	基数: 8↵ 八进制: 66238↵ 十进制: 3475	基数: 10↵ 十进制: 3475a↵ 十进制: 3475	基数: 16↵ 十六进制: d93g↵ 十进制: 3475

3) 判断任意进制数字

BOOL IsRDigit(char x, INTEGER r);

说明: r为基数。若字符x为*r*进制数字(仅包含允许的数字字符及大写或小写字母), 则函数值为1(真), 否则为0(假)。用下面的主函数来进行测试。

```
int main()
{
    INTEGER x, r;
    char c;
    printf("基数: ");
    scanf(ISF, &r);
    printf("数字: ", r);
    scanf("␣%c", &c);
    if (IsRDigit(c, r))
    {
        puts("Yes");
    }
    else
    {
        puts("No");
    }
    return 0;
}
```

测试用例:

基数: 2↵ 数字: 0↵ Yes	基数: 2↵ 数字: 1↵ Yes	基数: 8↵ 数字: 8↵ No	基数: 20↵ 数字: 5↵ Yes	基数: 20↵ 数字: J↵ Yes	基数: 20↵ 数字: K↵ No

4) 任意进制输入任意整数

BOOL ScanInteger(INTEGER *x, INTEGER r);

说明: r是基数。以r进制输入整数, 保存到指针x所指示的变量中。用下面的主函数来进行测试。

```
int main()
{
    INTEGER x, r;
    printf("基数: ");
    scanf(ISF, &r);
    printf(IPF "进制: ", r);
    ScanInteger(&x, r);
    printf("十进制: ");
```

```
    PrintInteger(x, 10);
    putchar('\n');
    return 0;
}
```

测试用例:

基数: 2↵ 2进制: 110110010011↵ 十进制: 3475	基数: 8↵ 8进制: -6623↵ 十进制: -3475	基数: 10↵ 10进制: 3475↵ 十进制: 3475	基数: 16↵ 16进制: -d93↵ 十进制: -3475
基数: 2↵ 二进制: 1101100100112↵ 十进制: 3475	基数: 8↵ 八进制: 66238↵ 十进制: 3475	基数: 10↵ 十进制: -3475a↵ 十进制: -3475	基数: 16↵ 十六进制: d93g↵ 十进制: 3475

10. 几何图形

创建MyIO.h和MyIO.c, 编写几何图形输出函数。

1) 一串字符

void Show(int number, char symbol);

说明: number为字符的数量, symbol为字符的编码。若number ≤ 0, 则不输出任何字符。用下面的主函数进行测试。

```
int main()
{
    int n;
    char s;
    printf("n = ? ");
    scanf("%d", &n);
    printf("s = ? ");
    scanf("␣%c", &s);
    Show(n, s);
    putchar('\n');
    return 0;
}
```

测试用例:

n = ? 8↵ s = ? *↵ ********	n = ? 5↵ s = ? #↵ #####	n = ? 0↵ s = ? !↵	n = ? -2↵ s = ? &↵

2) 直角三角形(上)

void ShowUpTrg(int height, char symbol);

说明: height为直角三角形的高(行数), symbol为组成直角三角形的字符。若height ≤ 0, 则不输出任何字符。用下面的主函数进行测试。

```
int main()
{
    int h;
    char s;
    printf("h = ? ");
    scanf("%d", &h);
    printf("s = ? ");
    scanf("␣%c", &s);
    ShowUpTrg(h, s);
    return 0;
}
```

测试用例:

h = ? 8↵ s = ? *↵ * ** *** **** ***** ****** *******	h = ? 5↵ s = ? #↵ # ## ### #### #####	h = ? 0↵ s = ? !↵	h = ? -2↵ s = ? &↵

********			

3) 直角三角形(下)

void ShowDnTrg(int height, char symbol);

说明: height为直角三角形的高(行数), symbol为组成直角三角形的字符。若height ≤ 0, 则不输出任何字符。用下面的主函数进行测试。

```
int main()
{
    int h;
    char s;
    printf("h = ? ");
    scanf("%d", &h);
    printf("s = ? ");
    scanf("␣%c", &s);
    ShowDnTrg(h, s);
    return 0;
}
```

测试用例:

h = ? 8↵ s = ? *↵ ********* ******** ******* ****** ***** **** *** ** *	h = ? 5↵ s = ? #↵ ##### #### ### ## #	h = ? 0↵ s = ? !↵	h = ? -2↵ s = ? &↵

4) 矩形

void ShowRect(int height, int width, char symbol);

说明: height为矩形的高(行数), width为矩形的宽(列数), symbol为组成直角三角形的字符。若height ≤ 0, 则不输出任何字符, 若width ≤ 0, 则只输出height行空白行。用下面的主函数进行测试。

```
int main()
{
    int h, w;
    char s;
    printf("h = ? ");
    scanf("%d", &h);
    printf("w = ? ");
    scanf("%d", &w);
    printf("s = ? ");
    scanf("␣%c", &s);
    ShowRect(h, w, s);
    return 0;
}
```

测试用例:

h = ? 4↵ w = ? 15↵ s = ? *↵ *************** *************** *************** ***************	h = ? 5↵ w = ? 20↵ s = ? #↵ #################### #################### #################### #################### ####################	h = ? 0↵ w = ? 20↵ s = ? !↵	h = ? 5↵ w = ? 0↵ s = ? #↵ ↵ ↵ ↵ ↵ ↵