C语言第五周作业-1

程序题

数字统计

> 【问题说明】用while循环语句编写程序，统计从键盘输入的数字字符出现的次数，并把其中的数字字符依次输出。
> 【输入形式】从键盘输入一串带数字字符的字符串。
> 【样例输入】 `shanghai12345china5678asian`
> 【样例输出】
> `1 2 3 4 5 5 6 7 8`
> `There are 9 digits!`
> 【样例说明】
> (1) 输入字符串以回车结束输入。
> (2) 分两行输出，第一行输出字符串中的数字，每个数字后面有一个空格，第二行输出结束后有换行符\n。

- 使用 `while` 循环的循环条件是 `输入不等于'\0'`
- 其实 `scanf()` 语句也可以放在 `while` 的条件里面 `scanf()` 语句使用返回值的，有输入为真，没有输入为假。
- 当然我会提供一个正常写法的版本

示例代码1

```c
#include <stdio.h>
int main()
{
    int d_i=0;
    char a,b[100];
    while(scanf("%c",&a)&&a!='\n'){
        if(a>='0'&&a<='9'){
            b[d_i]=a;
            d_i++;
        }
    }
    for (int i = 0; i < d_i; i++)
            printf("%c ",b[i]);
    printf("\nThere are %d digits!\n",d_i);
    return 0;
}
```

- 这里就是不断读入的一个 `while` 循环，而且将 `while` 与 `scanf` 嵌套使用

示例代码2

```
#include <stdio.h>
int main()
{
    int d_i=0;
    char a,b[100];
    scanf("c",&a);
    while(a!='\n'){
        if(a>='0'&&a<='9'){
            b[d_i]=a;
            d_i++;
        }
        scanf("%c",&a)
    }
    for (int i = 0; i < d_i; i++)
            printf("%c ",b[i]);
    printf("\nThere are %d digits!\n",d_i);\\这里是开始就换行，把光标移到下
一行再开始新的输入
    return 0;
}
```

- 这里在 `while` 循环前输入的原因是，让最开始的变量 `a` 有一个可供判断且能进入 `while` 循环的值，当然在定义的时候可以把比变量 `a` 给定一个不为 `\0` 的值。
- 或者我们可以使用 `do-while` 循环

```
#include <stdio.h>
int main()
{
    int d_i=0;
    char a,b[100];
    scanf("c",&a);
    do{
        scanf("%c",&a)
        if(a>='0'&&a<='9'){
            b[d_i]=a;
            d_i++;
        }
    }while(a!='\n')
    for (int i = 0; i < d_i; i++)
            printf("%c ",b[i]);
    printf("\nThere are %d digits!\n",d_i);\\这里是开始就换行，把光标移到下
一行再开始新的输入
    return 0;
}
```

计算位数并逆序输出

> 【问题说明】用do循环语句编写程序，输入一个5位或5位以下的正整数，逆序输出该数并计算它是几位数。
> 【输入形式】从键盘输入一个5位或5位以下的正整数。
> 【样例输入】 `Please input the number:3453`
> 【样例输出】
> `Inversed number is: 3543`
> `It has 4 bits.`
> 【样例说明】“Inversed number is: ”冒号为英文字符，冒号后面跟一个空格。输出后回车换行。

- 这是一个很明显进行字符操作的题目
- 如果不是题目要求一定要使用 `do-while` 循环，我们可以直接字符串读入。

示例代码

```c
#include <stdio.h>
int main()
{
    int d_i=0;
    char a,b[6];
    printf("Please input the number:");
    do{
        scanf("%c",&a);
        if(a>'0'&&a<'9'){
            b[d_i]=a;
            d_i++;
        }//依次输入，满足条件就存入数组中并计数
    }while (a!='\n');
    printf("Inversed number is: ");
    for (int i = d_i-1; i >= 0; i--)
        printf("%c",b[i]);
    printf("\nIt has %d bits.",d_i);
    return 0;
}
```

计算100以内奇数和

> 【问题说明】用for循环语句编写程序，求1~100间奇数之和，1+3+5+……+99。
> 【样例输出】sum=2500（输出结束有换行符）

- 这个没有好说的遍历1~100并加上判断就ok了
- 判断奇数的条件 `a%2!=0`

示例代码

```c
#include <stdio.h>
int main()
{
    int sum = 0;
    for (int i = 1; i <= 100; i++){
        if (i % 2 !=  0)//其实有一种更加简洁的位运算写法：i & 1
            sum += i;
    }
    printf("sum=%d\n", sum);
    return 0;
}
```

乘法表输出

> 【问题说明】输入 `n` ，计算 `n*n` 的乘法表。
> 【样例输出】每个结果用 `%4d` 的形式输出；输出所有结果后换行。
> 如输入 `4` ，则输出：
> 1 2 3 4
> 2 4 6 8
> 3 6 9 12
> 4 8 12 16

- 其实这里把乘法表想清楚就ok了，这里用这个4×4的例子说明

1	2	3	4
2	4	6	8
3	6	9	12
4	9	12	16

- 这里看起来就是两个循环嵌套就可以实现的，一行一行的输出：
  - 第一行 `1*1` `1*2` `1*3` `1*4`
  - 第二行 `2*1` `2*2` `2*3` `2*4`

示例代码

```
#include <stdio.h>
int main()
{
    int n;
    scanf("%d", &n);

    for (int i = 1; i <= n; i++){
        for (int j = 1; j <= n; j++)//第二个for循环，负责每一行的输出
            printf("%4d", i * j);
        printf("\n");//输出完一行以后换行进行下一次循环的下一行输出
    }
    return 0;
}
```

函数值计算1

> 【问题描述】求$1+\frac{1}{1+2}+\frac{1}{1+2+3}+\cdots+\frac{1}{1+2+3+\cdots+50}$的值
> 【输出形式】输出为sum=%f，输出后不换行。

- 这一道题唯一的难度就是在如何确定循环了
- 这里完成两个循环嵌套就ok,具体解析看注释

示例代码1(自定义函数版)

```
#include <stdio.h>
int Sum(int n)//用于计算第n项的分母值的函数
{
    int sum = 1;
    for (int i = 2; i <= n; i++)
        sum += i;
    return sum;
}

int main()
{
    double n=0;
    for (int i = 1; i <= 50; i++)
        n += 1.0 / (double)Sum(i);//返回值为int这里需要强制转换一下，也可将返回值改为double
    printf("sum=%f", n);
    return 0;
}
```

- 这里按照我自己的习惯写了一个自定义函数，当然是可以不用的。

示例代码2（无函数版）

```
#include <stdio.h>
int main()
{
    double sum=1.0;
    int i,t=1;
    for(i=2;i<=50;i++){
        t = t + i;
        sum += 1.0 / t;
    }
    printf("sum=%f",sum);
}
```

函数值计算2

> 【问题描述】输入n，求s=1+(1+20.5) + (1+20.5+30.5) +…+(1+20.5+30.5+…+n0.5)的和
> 【输出形式】输出为sum=六位小数的格式，输出后不换行。

- 就是通项公式：
  $a_1=0,a_n=1+20.5+.....+n*10+0.5=1+(2+.....+n)*10+(n-1)*0.5$
- 求a_n的和S_n的值
- 这个也是两个循环，第一个循环计算$a_1+a_2+.....+a_n$;第二个循环计算a_n值

示例代码

```
#include <stdio.h>
int main()
{
    int n;
    double sum=1;//默认存下a1的值
    scanf("%d", &n);
    for (int i = 2; i <= n; i++){//只要n>=2时才进入计算
        for (int j = 2; j <= i; j++)//第二个循环计算10*(2+3+……+n)
            sum += j*10.0;
        sum += (double)(i-1) * 0.5;
        sum ++;
    }
    printf("sum=%.6f", sum);
    return 0;
}
```

二分法求根

> 【问题描述】用二分法求方程$2x^3-4x^2+3x-6=0$在$(-10，10)$之间的根。
> 【输出形式】输出为6位小数，输出后不换行

- 其实就很简单了我们用二分法就可以了
- 用 `f(left)` 和 `f(right)` 和 `0` 比大小行了，保证 `0` 在两者之间
- 再用 `(left+right)/2` 代替 `left` 或者 `right` 中的一个就行
- 一个无限循环，退出条件为：$|f(x)-0|\leq 0.000001$

示例代码1（带函数版）

```c
#include <stdio.h>
#include <math.h>

double equation(double x)
{
    return 2 * pow(x, 3) - 4 * pow(x, 2) + 3 * x - 6;
}

double bisection(double left, double right)
{
    double epsilon = 1e-6;//限制精度为六位小数，即|f(x)-0|<0.000001
    double mid, fmid;

    while (right - left > epsilon)
    {
        mid = (left + right) / 2;
        fmid = equation(mid);

        if (fmid == 0)
            return mid;
        else if (fmid * equation(left) < 0)
            right = mid;
        else
            left = mid;
    }

    return (left + right) / 2;//返回结果
}

int main()
{
    double left = -10.0;
    double right = 10.0;
    double root = bisection(left, right);
    printf("%.6f", root);
    return 0;
}
```

示例代码2（无函数版本）

```c
#include <stdio.h>
#include <math.h>

int main()
{
    double left = -10.0;
    double right = 10.0;
    double epsilon = 1e-6;//限制精度为六位小数，即|f(x)-0|<0.000001
    double mid, fleft, fmid;
    while (right - left > epsilon)
    {
        mid = (left + right) / 2;
        fleft = 2 * pow(left, 3) - 4 * pow(left, 2) + 3 * left - 6;
        fmid = 2 * pow(mid, 3) - 4 * pow(mid, 2) + 3 * mid - 6;
        if (fmid == 0)
            break;
        if (fmid * fleft < 0)
            right = mid;
        else
            left = mid;
    }
    printf("%.6f", mid);
    return 0;
}
```