计算概论（C语言）习题课讲义06

内容概要

- 习题讲解
- 前五章梳理
- 课堂练习

习题讲解

哥德巴赫猜想的验证

算法框架：

```
int prime(int n); //判断输入n是否为素数
int goldbach(int n); //判断输入n是否可以分解为两个素数之和
```

同学代码点评一:

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

int isprime(int n){
    for(int m=2;m*m<=n;++m){
       if (n%m==0)
          return 0;}
    return 1;
}

int main(){
    for(;;){
        int a=0,t,n;double time1=0;
        printf("请输入需要分解的偶数，输入奇数结束程序：") ;
          scanf("%d",&n);
          time1=clock();
        if(n%2==0&&n>4){
            for(t=3;t<=n/2;t+=2){
                if (isprime(t)&&isprime(n-t)){
                    printf("%d=%d+%d\n",n,t,n-t);
                    a=a+1;
                }
             }
        printf("共有%d组,用时%f秒\n\n",a,(clock()-time1)/CLOCKS_PER_SEC);

        }
        else{
            printf("程序结束");
```

```
            return 0;
    }
    }
}
```

同学代码点评二:

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

int prime(int n){
 int m;
 for(m=2;m<=n;++m){
 if(n%m==0){
 break;
 }
 }
 if(m==n)
     return 1;
 else
     return 0;
}

int main(){
 int m,n,t;
 scanf("%d",&n);
 for(m=6;m<=n;m=m+2){
 for(t=3;t<=m-3;t=t+2){
 if (prime(t)==1 && prime(m-t)==1){
 printf("%d=%d+%d\n",m,t,m-t);
 break;
 }
 }
 if(t==m-3 && m>=8){
     printf("推翻了哥德巴赫猜想！！！！！！\n");
     break;
 }}
 if(m==n+2)
 printf("哥德巴赫猜想在小于等于%d成立\n",n);
    return 0;
}
```

C语言代码规范

虽然C语言的标准中并没有对代码的缩进和换行有硬性的要求，但一般大家都会有一些通俗的约定．这里推荐使用如下的代码缩进风格：

1. 条件语句和循环语句使用“{” 和 “}”，并且“{” 和 “}”各自独占一行；
2. 条件语句和循环语句中的执行语句缩进一级．缩进使用４空格．

```
//这里示范该种缩进风格
#include <stdio.h>
int main ()
{
   /* for 循环执行 */
   for( int a = 10; a < 20; a = a + 1 )
   {
      printf("a 的值： %d\n", a);
   }
   return 0;
}
```

同学代码点评三:

```
#include <stdio.h>
#include <math.h>
int isprime(n)
{
    int i=1;
    while(i< n-1)
    {
        i=i+1;
        if(n%i==0)
           return 0;
    }
        return 1;
}
int fact(n)
{
    int i=1,m=n;
    while(i<n-1)
    {
        i=i+1;
        m=m-i;
        if(isprime(i)&&isprime(m)==1)

            return 1;

    }
    return 0;

}
int judge(n)
{
    int p=4;
    while(p<=n)
    {
        p=p+2;
       printf("%d\n",fact(p));
    }

}
```

```
int main()
{
    judge(100);
    return 0;
}
```

输出所有素因子

代码点评一:

```
#include <stdio.h>
#include <math.h>
int isprime(n)
{
    int i=1;
    while(i<n-1)
    {
        i=i+1;
        if(n%i==0)
            return 0;

    }
    return 1;
}
void prt_pfactors(int n)
{
    int i=0;
    if(n>0)
    {
        while(i<n)
        {
        i=i+1;
        if(n%i==0)
        {
            if(isprime(i)!=0)
            {printf("%d\n",i);}
        }
        }
    }
    else if(n<0)
    {
        printf("%d\n",-1);
        n=-n;
        while(i<n)
        {
        i=i+1;
        if(n%i==0&&isprime(i)==1)
        {
            printf("%d\n",i);
        }
        }

    }
```

```
}

int main()
{
    int n;
    scanf("%d",&n);
    prt_pfactors(n);
    return 0;
}
```

代码点评二:

```
#include <stdio.h>

void prt_pfactors(int n){
    int m;
    if(n>0){
        for(m=2;m<=n;++m){
            if(n%m==0){
                printf("%d\n",m);
                prt_pfactors(n/m);
                break;
            }
        }
    }
    else{
        printf("-1\n");
            prt_pfactors(-n);
    }

}

int main () {
    int n;
    scanf("%d",&n);
    prt_pfactors(n);
    return 0;
}
```

代码点评三:

```
#include <stdio.h>
int i=2;
void prt_pfactors(int n){
    if(n<0){printf("-1\n");
            n=-n;}
    while(n!=1){
        while((n%i==0)){
            printf("%d\n",i);
            n=n/i;
        }
        i++;
```

```
    }
}
int main () {
    int n;
    scanf("%d",&n);
    prt_pfactors(n);
    return 0;
}
```

使用级数求解pi近似值

代码点评一:

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

int main()
{
        double sum=0;
        double exact=3.14159265;
        double pi=0;
        int i=1;
        while(exact-pi>1e-5)
        {
                sum+=1.0/i/i;
                i++;
                pi=sqrt(6.0*sum);
        }
        printf("%.6f %.6f %d\n",sum,pi,i-1);

        pi=0;
        sum=0;
        i=1;
        while(exact-pi>1e-6)
        {
                sum+=1.0/i/i;
                i++;
                pi=sqrt(6.0*sum);
        }
        printf("%.7f %.7f %d\n",sum,pi,i-1);

        pi=0;
        sum=0;
        i=1.0;
        while(exact-pi>1e-7)
        {
                sum+=1.0/i/i;
                i++;
                pi=sqrt(6.0*sum);
        }
        printf("%.20f %.20f %d\n",sum,pi,i-1);

        long double pi_l=0;
```

```
        long double sum_l=0;
        i=1.0;
        while(exact-pi_l>1e-7)
        {
                sum_l+=1.0/i/i;
                i++;
                pi_l=sqrtl(6.0*sum_l);
        }
        printf("%.20Lf %.20Lf %d\n",sum_l,pi_l,i-1);
        return 0;
}
```

代码点评二:

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

int main()
{
   long long n = 1;
   long double sum = 0, pi = 3.14159265;
   while (pi - sqrtl(6 * sum) > 0.00001)
   {
      sum = sum + 1. / (n * n);
      n++;
   }
   printf("%.6lf %.6lf %lld\n", sum, sqrtl(6 * sum), n - 1);
   while (pi - sqrtl(6 * sum) > 0.000001)
   {
      sum = sum + 1. / (n * n);
      n++;
   }
   printf("%.7lf %.7lf %lld\n", sum, sqrtl(6 * sum), n - 1);
   while (pi - sqrtl(6 * sum) > 0.0000001)
   {
      sum = sum + 1. / (n * n);
      n++;
   }
   printf("%.8lf %.8lf %lld\n", sum, sqrtl(6 * sum), n - 1);
   return 0;
}
```

代码点评三:

```
#include <stdio.h>
#include <math.h>
double pi=3.14159265;

void pi_app(double error1,double error2,double error3){
    int i=0;
    long double sum=0.,p=0;
    double e=1.;
```

```
    while(e>=error1){
        i++;
        sum=sum+1./(i*i);
        p=sqrtl(6.*sum);
        e=fabs(p-pi);
    }
    printf("%.7Lf %.7Lf %d",sum,p,i);
    while(e>=error2){
        i++;
        sum=sum+1./(i*i);
        p=sqrtl(6.*sum);
        e=fabs(p-pi);
    }
    printf("%.7Lf %.7Lf %d",sum,p,i);
    while(e>=error3){
        i++;
        sum=sum+1./(i*i);
        p=sqrtl(6.*sum);
        e=fabs(p-pi);
    }
    printf("%.7Lf %.7Lf %d",sum,p,i);
    return;
}
int main(){
    pi_app(0.00001,0.000001,0.0000001);
    return 0;
}
```

完全数的判断

代码点评一:

```
#include<stdio.h>

int perfect_or_not(int n)
{
  int sum=0;
  int m;
  for(m=1;m<=n;++m)
  {
    if(n%m==0)
    {
      sum = sum + m;
    }
    else{}
  }
  if((sum-2*n)>0) return 1;
  else if((sum-2*n)==0) return 0;
  else return -1;
}

void perfect(int n)
{
```

```
  int m;
  printf("1\n");
  for(m=1;m<=n;++m)
  {
    if (perfect_or_not(m)==0) printf("%d\n",m);
    else {}
  }
}


void class(int a,int b)
{
  int m=0,n=0,p=0,t;
  for(t=a;t<=b;++t)
  {
    if(perfect_or_not(t)==-1){m=m+1;}
    else if(perfect_or_not(t)==0){n=n+1;}
    else{p=p+1;}
  }
  printf("from %d to %d: %d %d %d\n",a,b,m,n,p);
}

int main()
{
  int n;
  perfect(1000);
  for(n=1001;n<10000;n=n+1000)
  {class(n,n+999);}
  for(n=10001;n<100000;n=n+10000)
  {class(n,n+9999);}
  return 0;
}
```

代码点评二:

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

int isperfect(long n)
{
    int i = 2, sum = 0;
    while (i <= sqrt(n))
    {
        if (n%i == 0 && i != sqrt(n))
            sum = sum + i + n / i;
        else if (n%i == 0 && i == sqrt(n))
            sum = sum + i;
        i++;
    }
    sum = sum + 1;
    return sum == n ? 0 : (sum < n ? -1 : 1);
}
```

```c
int main()
{
    long n, m, a=0, b=0, c=0;
    for (n = 0; n <= 1000; n++)
    {
        if (isperfect(n) == 0)
            printf("%d\n", n);
    }
    for (n = 1000; n <= 9000; n = n + 1000)
    {
        for (m = n + 1; m <= n + 1000; m++)
        {
            if (isperfect(m) == -1)
                a = a + 1;
            else if (isperfect(m) == 0)
                b = b + 1;
            else
                c = c + 1;
        }
        printf("from %d to %d: %d %d %d\n", n + 1, n + 1000, a, b, c);
        a = 0, b = 0, c = 0;
    }
    for (n = 10000; n <= 90000; n = n + 10000)
    {
        for (m = n + 1; m <= n + 10000; m++)
        {
            if (isperfect(m) == -1)
                a = a + 1;
            else if (isperfect(m) == 0)
                b = b + 1;
            else
                c = c + 1;
        }
        printf("from %d to %d: %d %d %d\n", n + 1, n + 10000, a, b, c);
        a = 0, b = 0, c = 0;
    }
    return 0;
}
```

前五章梳理

第一章

1. 从C语言源文件到可执行文件的加工流程?
2. 名词解释: IDE, 编译器, gcc

第二章

1. 合法的标识符?
2. 为什么要在使用`printf`函数时,使用stdio.h文件?
3. 解释类型转化描述符`%d`,`%x`,`%f`,`%e`,`%.8f`
4. 计算如下表达式子的结果`(int)(3.6/2)+5/2+01+0x1`

第三章和第四章

1. 运行语句`x=1?x==2:x=3`,x值为?
2. `break`和`continue`语句作用是？
3. Fibonacci数列的递归计算?循环计算?
4. 求最大公约数的递归方式?循环方式?
5. 使用递归的思想分析Hanoi塔问题的复杂度.

第五章

1. 字符' 0 '和int 0的区别
2. `if(c >= 'a' && c <= 'z')`的作用是？
3. `switch`语句中的`case`语句为什么一般以break结尾？
4. 全局变量和局部变量的区别？
5. 解释宏定义`#define pi 3.14159`; 如果源程序中定义函数pie,会怎样？

随机数的生成

计算机实际上无法生成真正的随机数，只能生成伪随机数.

使用`int rand(void)`函数产生随机整数，需要包含`stdlib.h`的头文件.

使用`void srand(unsigned seed)`设定随机数种子.

演示：随机数种子的作用

课堂练习

- (P103 20)　写一个程序，它输出所读入的一系列整数的平均值，假定输入的第一个整数表示数据的个数.

- (P103 21)　假设程序由输入得到的一系列正实数是一条折线在x=0,1,2...时的对应值，输入负数时结束输入.求该折线和 x 轴之间区域的面积.

- (P142 2)　使用字符分类函数写一个函数，统计输入中的十进制和十六进制整数字符．输入以EOF结束.如果不用字符分类函数呢？

- (P142 5)　使用字符输出画一个实心的圆；空心的呢？