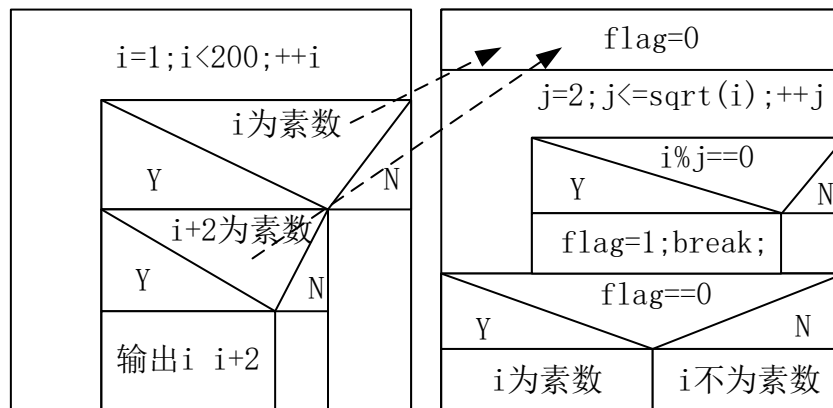


# 作业 6

## 必做题

### 第一题

流程图/NS 图：



设计算法：

对 1~199 中各整数进行遍历，每个整数判断它和它加二的数是否均为素数，若是，则是孪生素数对，反之则不是；

判定为素数的算法：对小于待测数的平方根的非 1 正整数逐一遍历，若均不能使待测数被整除，则是素数；若待测数可被任何一个整除，则非素数。

程序代码：

```

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define PRIME 1
#define COMPOSITION 0
char cPrimeJudgement(int);
int main() {
    int iTest=0;
    printf("10至200之间的孪生素数对有：\n"); /*首行提示输出*/
    for (iTest=10; iTest<=199; ++iTest) {
        switch(cPrimeJudgement(iTest)+cPrimeJudgement(iTest+2)) { /*仅当两数均为
        素数时才为孪生素数对，此时和为2*/
            case PRIME+PRIME: printf("%d与%d\n", iTest, iTest+2); /*是孪生素数对则
            输出*/
        }
    }
    system("pause");
    return 0;
}

```

/\*判定一个整数是否为素数的函数\*/

```

char cPrimeJudgement(int iNumberToBeJudged) {
    char cFlag=COMPOSITION;
    int iRound=0;

```

```

    for (iRound=2;iRound<=(int) sqrt((double) iNumberToBeJudged);++iRound) { /*逐一检测小于其平方根的整数*/
        if (iNumberToBeJudged%iRound==0) { /*若能被整除，则不是素数*/
            cFlag=COMPOSITION;
            break;
        }
        else cFlag=PRIME; /*若不能被整除，则有是素数的可能*/
    }
    switch (cFlag) {
        case COMPOSITION: return COMPOSITION; /*不是素数*/
        case PRIME: return PRIME; /*是素数*/
    }
}

```

运行结果：

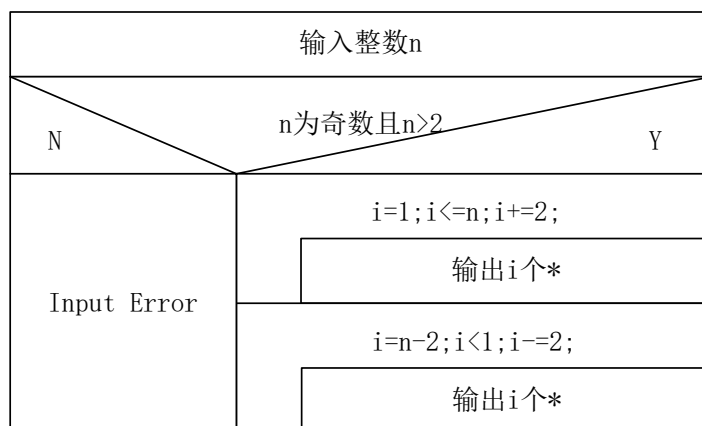
```

10至200之间的孪生素数对有：
11与13
17与19
29与31
41与43
59与61
71与73
101与103
107与109
137与139
149与151
179与181
191与193
197与199
请按任意键继续 . . . |

```

## 第二题

流程图/NS 图：



设计算法：

先判定 n 是否是有效输入，如果无效就输出 “Input Error”，如果有效则打印菱形。  
打印菱形按行进行，前  $(n+1)/2$  行从 1 个开始每行增加两个\*；后  $(n-1)/2$  行从对角线开始每行减少两个\*。

程序代码：

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
char cPrintDiamond(int);
int main() {
    int iNumber=0;
    printf("请输入一个大于2的奇数作为菱形的对角线长度：");
    scanf("%d",&iNumber);
    if (iNumber>2 && iNumber%2) cPrintDiamond(iNumber);
    else printf("Input Error");
    system("pause");
    return 0;
}

char cPrintDiamond(int iDiagonal) {
    int iRow;/*一行需要打印的*个数*/
    int iSpace;/*一行需要打的空格的次序*/
    int iPrint;/*一行需要打的*的次序*/
    for (iRow=1;iRow<=iDiagonal;iRow+=2) { /*上半部分*/
        for (iSpace=0;iSpace<(iDiagonal-iRow)/2;++iSpace) printf(" "); /*打印空格*/
        for (iPrint=0;iPrint<iRow;++iPrint) printf("*"); /*打印***/
        printf("\n"); /*行末换行*/
    }
    for (iRow=iDiagonal-2;iRow>0;iRow-=2) { /*下半部分*/
        for (iSpace=0;iSpace<(iDiagonal-iRow)/2;++iSpace) printf(" "); /*打印空格*/
        for (iPrint=0;iPrint<iRow;++iPrint) printf("*"); /*打印***/
        printf("\n"); /*行末换行*/
    }
    return 0;
}
```

运行结果：

```
请输入一个大于2的奇数作为菱形的对角线长度：100
Input Error请按任意键继续. . . |
```

```

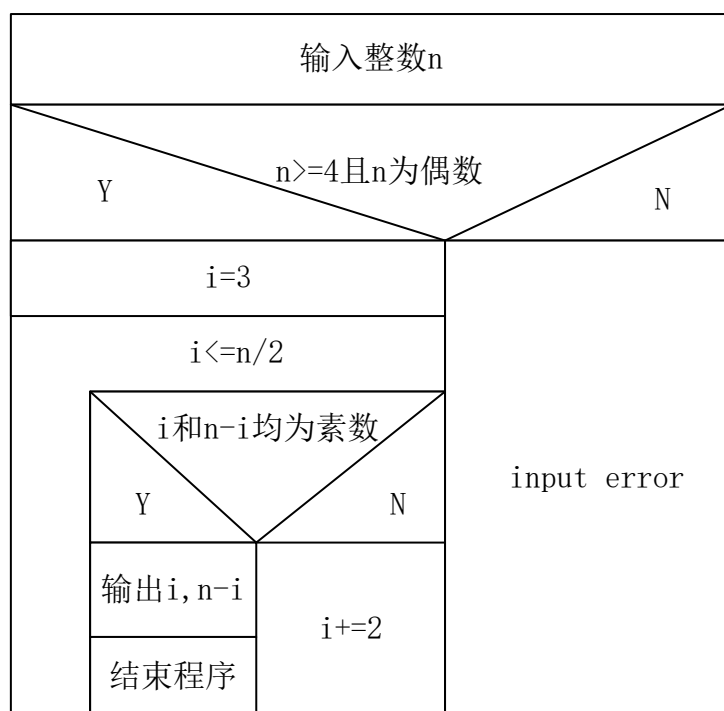
      *
     **
    ***
   ****
  *****
 *****
*****
 *****
  *****
   ****
    ***
     **
      *

```

请按任意键继续. . .

### 第三题

流程图/NS 图:



算法设计:

先判断输入的整数  $n$  是否有效，若无效，输出“input error”；若有效，遍历  $3 \sim n/2$ ，找到最小的能满足条件的  $p_1$ ,  $p_2 = n - p_1$ 。满足条件的判断即  $p_1$  和  $p_2$  均为素数，这个可以通过第一题的算法进行判断，即考虑一个待测数，遍历  $3 \sim$  待测数的平方根的所有整数，若都不能整除待测数，则待测数为素数。

程序代码：

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define prime 1
#define not_prime 0
```

```

#define test(a) cJudgePrime(a, (long long)sqrt((double)a)+1) /*判断是否为质数的程序的简写*/
char cJudgePrime(long long, long long); /*变量声明*/
int main() {
    long long llN; /*输入整数存储的变量*/
    printf("请输入一个大于4的偶数: ");
    scanf("%lld", &llN);
    if (llN>=4 && llN%2==0) {
        long long llTest=3; /*对3开始的奇数进行判断尝试*/
        for (; llTest<=llN/2; llTest+=2) {
            if (test(llTest) && test(llN-llTest)) {
                printf("%lld=%lld+%lld", llN, llTest, llN-llTest);
                break;
            }
        }
    }
    else printf("input error"); /*输入错误*/
    system("pause");
    return 0;
}

/*递归判断是否为质数的函数*/
char cJudgePrime(long long llCandidate, long long llDivision) {
    if (llCandidate%llDivision==0) return not_prime; /*初始条件, 发现能被整除*/
    else if (llDivision==2) return prime; /*初始条件, 发现没有能整除的数*/
    else { /*暂时还没发现能整除的数, 向下递归*/
        if (cJudgePrime(llCandidate, llDivision-1)==prime) return prime;
        else return not_prime;
    }
}

```

运行结果:

```

请输入一个大于4的偶数: 77
input error请按任意键继续. .

```

```

请输入一个大于4的偶数: 123456
123456=7+123449请按任意键继续. . .

```

```

请输入一个大于4的偶数: 131072
131072=13+131059请按任意键继续. . .

```

## 选做题

### 第一题

程序代码:

```

#include<stdio.h>
#include<stdlib.h>
int main() {
    unsigned short usAppleAmount=fApple(1);
    char cFor;
    printf("原来共有%hu个苹果。\\n",usAppleAmount);
    /*打印每个孩子分到的苹果数*/
    for (cFor=1;cFor<=4;cFor++) {
        printf("第%d个小孩分到%hu个苹果。\\n",cFor,(int)((double)(usAppleAmount+1)/(double)(cFor+1)));
        usAppleAmount-=(int)((double)usAppleAmount+1)/(double)(cFor+1));
    }
    printf("第5个小孩分到%hu个苹果。\\n",usAppleAmount);
    system("pause");
    return 0;
}
/*计算一开始有多少苹果的递归函数*/
int fApple(char cRound) {
    if (cRound==5) return 11;
    else return (int)((double)fApple(cRound+1)*(cRound+1)/(double)(cRound)+1);
}

```

运行结果:

```

原来共有59个苹果。
第1个小孩分到30个苹果。
第2个小孩分到10个苹果。
第3个小孩分到5个苹果。
第4个小孩分到3个苹果。
第5个小孩分到11个苹果。
请按任意键继续. . . |

```

## 第二题

程序代码:

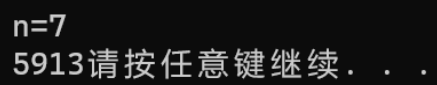
```

#include<stdio.h>
#include<stdlib.h>
int main() {
    short sNumber;
    char cTempMain;
    long long llSum=0;
    printf("n=");
    scanf("%hd",&sNumber);
    /*求和*/
    for (cTempMain=1;cTempMain<=sNumber;cTempMain++)
        llSum+=iJieCheng(cTempMain);
}

```

```
printf("%lld", llSum);  
system("pause");  
return 0;  
}  
/*计算阶乘*/  
int iJieCheng(char cNum) {  
    char cTemp;  
    long long i64JieCheng=1;  
    for (cTemp=1; cTemp<=cNum; cTemp++) i64JieCheng*=cTemp;  
    return i64JieCheng;  
}
```

运行结果：



```
n=7  
5913请按任意键继续. . .
```