实验八　结构体

*1、定义一个“点”结构体，输入二个点的值，然后计算二点间的距离。

```
#include<stdio.h>
#include<math.h>
struct point       //这个结构体类型是所有点都具备的特性
{
    double x,y;
};
main()
{   struct point p1,p2;  //p1,p2 这二个点是具体实际的点
    double dis;
    printf("请输入第一个点的值:");
    scanf("%lf%lf",&p1.x,&p1.y);
    printf("请输入第二个点的值:");
    scanf("%lf%lf",&p2.x,&p2.y);
    dis=sqrt((p1.x-p2.x)*(p1.x-p2.x)+(p1.y-p2.y)*(p1.y-p2.y));
    printf("(%lf,%lf)与(%lf,%lf)的距离是：%lf\n",p1.x,p1.y,p2.x,p2.y,dis);
}
```

**2、某班有 40 个学生，每个学生的信息包含有学号、姓名、性别、语文成绩、数学成绩、英语成绩、总分。从键盘输入 40 个学生的信息，计算总分，然后按总分从高到低顺序输出。

```
#include<stdio.h>
#include<string.h>
#define N 10
struct student
{   char no[10];
    char name[20];
    char sex;
    int chinese,math,english,total;
};
void printstu(struct student s)
{
    printf("%12s%9s%3c%5d%5d%5d%5d\n",s.no,s.name,s.sex,s.chinese,s.m
ath,s.english,s.total );
}
void sort(struct student s[N])
{   int i,j;
    struct student t;
    for(i=N-1;i>0;i--)
      for(j=0;j<i;j++)
        if(s[j].total <s[j+1].total )
```

```
        {
            t=s[j];
            s[j]=s[j+1];
            s[j+1]=t;
        }
//注意：结构体变量之间可以整体直接赋值，实现结构变量的交换很方便
main()
{
    struct student s[N];
    int i;
    for(i=0;i<N;i++)
    {

    scanf("%s%s %c%d%d%d",s[i].no,s[i].name,&s[i].sex,&s[i].chinese,&
s[i].math,&s[i].english);
        s[i].total=s[i].chinese +s[i].math +s[i].english;
    }
    printf("排序前：\n");
    for(i=0;i<N;i++)
        printstu(s[i]);
    sort(s);
    printf("排序后：\n");
     for(i=0;i<N;i++)
        printstu(s[i]);
}
```

输入以下数据：

202120001　王林　　f　89 80 78

202120002　李森　　m　99 87 84

202120003　张小平　f　78 70 90

202120004　王伟　　f 81 82 78

202120005　李国　　m　99 77 84

202120006　张华　　m　78 75 98

202120007　李宁　　f　83 87 78

202120008　吴天　　f　79 87 64

202120009　张小平　f　78 75 90

202120010　高小林　f　89 80 78

排序前：

　202120001　　王林　f　　89　　80　　78　247

202120002	李森	m	99	87	84	270
202120003	张小平	f	78	70	90	238
202120004	王伟	f	81	82	78	241
202120005	李国	m	99	77	84	260
202120006	张华	m	78	75	98	251
202120007	李宁	f	83	87	78	248
202120008	吴天	f	79	87	64	230
202120009	张小平	f	78	75	90	243
202120010	高小林	f	89	80	78	247

排序后：

202120002	李森	m	99	87	84	270
202120005	李国	m	99	77	84	260
202120006	张华	m	78	75	98	251
202120007	李宁	f	83	87	78	248
202120001	王林	f	89	80	78	247
202120010	高小林	f	89	80	78	247
202120009	张小平	f	78	75	90	243
202120004	王伟	f	81	82	78	241
202120003	张小平	f	78	70	90	238
202120008	吴天	f	79	87	64	230

**3、定义一个“日期”结构体（包括年、月、日），编程从键盘输入年(2018)、月、日，然后计算并输出该日是星期几。注意本题一定要用结构体实现，2018年1月1号是星期一。

```
#include<stdio.h>
struct date
{       int year,month,day;
};
int  leap(int year)
{        if(year%4==0&&year%100!=0||year%400==0)
                return 1;
        else
                return 0;
}
```

```
int main()
{       struct date d;
        int totalday=0;
        scanf("%d%d%d",&d.year,&d.month,&d.day);
        switch(d.month-1)
        {
                case 11:totalday+=30;
                case 10:totalday+=31;
                case 9:totalday+=30;
                case 8:totalday+=31;
                case 7:totalday+=31;
                case 6:totalday+=30;
                case 5:totalday+=31;
                case 4:totalday+=30;
                case 3:totalday+=31;
                case 2:if(leap(d.year)) totalday+=29;
                                else    totalday+=28;
                case 1:totalday+=31;
                case 0:totalday+=d.day;
        }
        switch(totalday%7)
        {
                case 0:printf("星期日");break;
                case 1:printf("星期一");break;
                case 2:printf("星期二");break;
                case 3:printf("星期三");break;
                case 4:printf("星期四");break;
                case 5:printf("星期五");break;
                case 6:printf("星期六");break;
        }
}
```

**4、如有定义 struct node

```
{   int no;
    struct node *next;
};
```

1）建立一个单向链表，链表共有 10 个结点，结点中的 no 值从 1 到 10 顺序给出。2）能删除一个结点。3）能插入一个结点。

```
#include<stdio.h>
#include<malloc.h>
struct node
{   int no;
    struct node* next;
};
```

```
struct node* create(int n)   //n 为建立的结点数
{    int i;
     struct node *h=NULL,*last,*p;
//以下建立链表的方式与课堂上有所不同，在这里第一个结节也是在循环中实现的。
     for(i=1;i<=n;i++)
     {
          p=(struct node*)malloc(sizeof(struct node));
          p->no=i;p->next=0;
          if(h==NULL)   //如果是建立第一个结点
               h=p;
          else               //如果建立的不是第一个结点
               last->next=p;
          last=p;
     }
     return h;
}
struct node* del(struct node *h,int x)
{    struct node *p1=h,*p2;
     if(h->no==x)   //如果是删除头结点
          h=h->next;
     else
     {
          while(p1->no!=x&&p1->next!=NULL)
          {
               p2=p1;
               p1=p1->next;
          }
          if(p1->next!=NULL||p1->no==x)
                    //如果 p1 不是最后一个结点，或者最后一个结点是要删除的结点
               p2->next=p1->next;
                    //当这个结点值找不到时，不用删除
     return h;
}
struct node * ins(struct node *h,int x)
{
     struct node *p,*p1=h,*p2;
     p=(struct node*)malloc(sizeof(struct node));
     p->no=x;
     if(h->no>x)        //如果是插在头结点前
     {
          p->next=h;
           h=p;
     }
     else
```

```
    {
        while(p1->no<x&&p1->next!=NULL)
        {
            p2=p1;
            p1=p1->next;
        }
        if(p1->no>x)   //如果不是插在最后一个结点之后
            p2->next=p,p->next=p1;
        else
            p1->next=p,p->next=NULL;
    }
    return h;
}
void printlink(struct node* h)
{
    struct node *p=h;
    while(p!=NULL)
    {
        printf("%5d",p->no);
        p=p->next;
    }
    putchar('\n');
}
main()
{
    struct node *h;
    h=create(10);
    printlink(h);
    int i;
    h=del(h,1);   //删除头结点
    printlink(h);
     h=del(h,10); //删除最后一个结点
    printlink(h);
    h=del(h,5);   //删除中间某个结点
    printlink(h);
     h=del(h,11);   //删除不存在的结点
    printlink(h);
     h=ins(h,1);     //插入头结点
    printlink(h);
    h=ins(h,10);   //插入到最后
    printlink(h);
    h=ins(h,5);
    printlink(h);//插入在中间某一位置
}
```

***5、将第 2 题用链表来实现。

```
#include<stdio.h>
#include<malloc.h>
typedef struct _stu
{     char stuno[10],name[10],gender;
      double score1,score2,score3,total;
      struct _stu *next;
} stu;
main()
{     int i;
      stu *head, *p2, *temp;
      head=(stu *)malloc(sizeof(stu));
      head->total=0x7fffffff;//这个结点永远在第一，所以值为最大
      head->next=NULL;
      //第一个结点不会参加排名，设这个结点的主要目的是为了插入与删除的方便。
      for (i=1; i<=40; i++)
      {     p2=(stu *)malloc(sizeof(stu)); //新的结点
            scanf("%s%s %c%lf%lf%lf",p2->stuno,p2->name,&p2->gender,
                        &p2->score1,&p2->score2,&p2->score3);
            p2->total = p2->score1 + p2->score2 + p2->score3;
            //输入完数据计算出总分，根据总分值插入到适当位置
            temp=head;
            while(temp->next!=NULL)
            {
                  if (temp->total >= p2->total && p2->total >= temp->next->total) break;
                  temp=temp->next;
            }
            //循环直至 p2->total 的值在 temp 所指结点与 temp->next 所指结点之间
            p2->next=temp->next;
            temp->next=p2;
            //以上二句是将新结点插入适当的位置，
            //p2=NULL;  这句话有用吗？
      }
      temp=head->next;
      printf("排名     学号            姓名        总分\n" );
      i=1;
      while(temp!=NULL)
      {
            printf("%3d%11s%11s%6.0lf\n",i++,temp->stuno,temp->name,temp->total);
            temp=temp->next;
      }
}
```