

```
1 #include <stdio.h>
2 #define N 10
3
4
5 void inputArray(int a[], int n)
6 {
7     for (int i=0; i<n; i++)
8         scanf ("%d", &a[i]);
9 }
10
11
12 int sequenceSearch(int a[], int value, int n)
13 {
14     int i, index=-1;
15
16     for (i=0; i<n; i++)
17         if (a[i]==value) {
18             index = i;
19             break;
20         }
21
22     return index;
23 }
24
25
26 int main()
27 {
28     int arr[N], num, result;
29
30     // printf("Input %d Numbers:\n", N);
31     inputArray(arr, N);
32     // putchar('\n');
33     // printf("Input search number:\n");
34     scanf ("%d", &num);
35
36     result = sequenceSearch(arr, num, N);
37     if (result != -1)
38         printf ("num is arr[%d]: %d", result, arr[result]);
39     else
40         printf ("num does not exist.");
41
42     return 0;
43 }
44
```

```

1 #include <stdio.h>
2
3 #define MaxNum 20
4 #define StrLen 100
5
6
7 void udfGetString(char s[], int maxlen)
8 {
9     int i=0;
10    char c;
11
12    while ((c=getchar()) != '\n' && i<maxlen-1)
13        s[i++] = c;
14    s[i] = '\0';
15 }
16
17
18 int udf_strlen(char s[])
19 {
20     int i;
21
22     for (i=0; s[i]; i++);
23     return i;
24 }
25
26
27 void udf_sort(char s[][StrLen], int num)
28 {
29     int i, j, k;
30     char tmp[StrLen];
31
32     for (i=0; i<num-1; i++) {
33         k = i;
34         for (j=i+1; j<num; j++)
35             if (udf_strlen(s[k])>udf_strlen(s[j]))
36                 k = j;
37         for (j=0; tmp[j]=s[i][j]; j++);
38         for (j=0; s[i][j]=s[k][j]; j++);
39         for (j=0; s[k][j]=tmp[j]; j++);
40     }
41 }
42
43
44 void udf_print(char s[][StrLen], int num)
45 {
46     for (int i=0; i<num; i++)
47         puts(s[i]);
48 }
49
50
51 int main()
52 {
53     char s[MaxNum][StrLen];
54     int i, num;
55
56     scanf ("%d", &num);
57     getchar();
58     for (i=0; i<num; i++)
59         udfGetString(s[i], StrLen);
60
61     udf_sort(s, num);
62     udf_print(s, num);
63
64     return 0;
65 }
66

```

```
1 #include <stdio.h>
2
3 #define M 5
4 #define N 6
5
6
7 void merge(int a[], int m, int b[], int n, int c[])
8 {
9     int i, j, x;
10
11    for (i=0; i<m+n; i++) {
12        x = (i<m) ?a[i]:b[i-m];
13        for (j=i; j>0; j--)
14            if (c[j-1]>x)
15                c[j] = c[j-1];
16            else
17                break;
18        c[j] = x;
19    }
20}
21
22
23 int main()
24 {
25     int a[M], b[N], c[M+N];
26     int i;
27
28     for (i=0; i<M; i++)
29         scanf ("%d", &a[i]);
30     for (i=0; i<N; i++)
31         scanf ("%d", &b[i]);
32
33     merge(a, M, b, N, c);
34
35     for (i=0; i<M+N; i++)
36         printf ("%d ", c[i]);
37
38     return 0;
39 }
40
```

```

1 #include <stdio.h>
2 #define N 6
3
4 typedef struct date {
5     int year;
6     int month;
7 } DATE;
8
9 struct book {
10     int num;
11     char title[20];
12     DATE ptime;
13 };
14
15 void sortBook(struct book[], int);
16
17
18 int main()
19 {
20     struct book lib[N];
21     int i;
22
23     for (i=0; i<N; i++)
24         scanf ("%d%s%d%d", &lib[i].num, lib[i].title, &lib[i].ptime.year,
25             &lib[i].ptime.month);
26     sortBook(lib, N);
27     for (i=0; i<N; i++)
28         printf ("%d %s %d %d\n", lib[i].num, lib[i].title,
29             lib[i].ptime.year, lib[i].ptime.month);
30
31
32
33 void sortBook(struct book lib[], int n)
34 {
35     int i, j, k;
36     struct book temp;
37
38     for (i=0; i<n-1; i++) {
39         k = i;
40         for (j=i+1; j<n; j++)
41             if (lib[k].ptime.year>lib[j].ptime.year)
42                 k = j;
43             else if(lib[k].ptime.year==lib[j].ptime.year &&
44                 lib[k].ptime.month>lib[j].ptime.month)
45                 k = j;
46             temp = lib[i];
47             lib[i] = lib[k];
48             lib[k] = temp;
49     }
50 }
51

```

```

1 #include <stdio.h>
2 #include <math.h>
3 #include <string.h>
4
5 #define PI 3.141592654
6
7 struct point {
8     double x;
9     double y;
10};
11
12
13 void translation(struct point pt[], double tl_x, double tl_y, int num)
14 {
15     for (int i=0; i<num; i++) {
16         pt[i].x += tl_x;
17         pt[i].y += tl_y;
18     }
19 }
20
21
22 void scale(struct point pt[], double s_x, double s_y, int num)
23 {
24     for (int i=0; i<num; i++) {
25         pt[i].x *= s_x;
26         pt[i].y *= s_y;
27     }
28 }
29
30
31 void rotation(struct point pt[], double angle, int num)
32 {
33     double a[2][2];
34     struct point temp;
35
36     angle = angle * PI / 180;
37     a[0][0] = cos(angle);
38     a[0][1] = -sin(angle);
39     a[1][0] = sin(angle);
40     a[1][1] = cos(angle);
41
42     for (int i=0; i<num; i++) {
43         temp.x = pt[i].x;
44         temp.y = pt[i].y;
45         pt[i].x = temp.x*a[0][0]+a[0][1]*temp.y;
46         pt[i].y = temp.x*a[1][0]+a[1][1]*temp.y;
47     }
48 }
49
50
51 int main()
52 {
53     int i=0, num=0;
54     char mode, action[10];
55     double angle, tl_x, tl_y, s_x, s_y;
56     struct point pt[10];
57
58     do {
59         printf("请输入坐标个数(>=2): ");
60         scanf("%d", &num);
61     } while (num<2);
62
63     for (i=0; i<num; i++) {
64         printf("请输入【第%d个】点的横x、纵y坐标: ", i+1);
65         scanf("%lf%lf", &pt[i].x, &pt[i].y);
66     }
67
68     do {
69         getchar();
70         printf("请选择处理方式: 平移(t)、缩放(s)、旋转(r): ");
71         mode = getchar();
72     } while (mode!='t' && mode!='s' && mode!='r');
73
74     switch (mode) {
75     case 't':
76         printf("请输入水平及垂直的平移量: ");
77         scanf("%lf%lf", &tl_x, &tl_y);
78         translation(pt, tl_x, tl_y, num);
79         strcpy(action, "平移");
80         break;
81
82     case 's':
83         printf("请输入水平及垂直的缩放比例: ");
84         scanf("%lf%lf", &s_x, &s_y);
85         scale(pt, s_x, s_y, num);
86         strcpy(action, "缩放");
87         break;
88
89     case 'r':
90         printf("请输入旋转角度: ");
91         scanf("%lf", &angle);
92         rotation(pt, angle, num);
93         strcpy(action, "旋转");
94         break;
95     }
96
97     printf("经过【%s】处理后, 坐标值如下: \n", action);
98     for (i=0; i<num; i++)
99         printf("%f %f\n", pt[i].x, pt[i].y);
100
101     return 0;
102 }
103

```

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAXOP    100
6 #define NUMBER   '0'
7 #define MAXVAL   100
8 #define BUFSIZE  100
9
10 void    push(double);
11 double  pop(void);
12 int     getop(char[]);
13 int     getch(void);
14 void    ungetch(int);
15
16 int     sp=0;
17 double  val[MAXVAL];
18 int     bufp=0;
19 char    buf[BUFSIZE];
20
21
22 int main()
23 {
24     int     type;
25     double op2;
26     char   s[MAXOP];
27
28     while ((type=getop(s))!=EOF) {
29         switch (type) {
30             case NUMBER:
31                 push(atof(s));
32                 break;
33             case '+':
34                 push(pop() + pop());
35                 break;
36             case '*':
37                 push(pop() * pop());
38                 break;
39             case '-':
40                 op2 = pop();
41                 push(pop() - op2);
42                 break;
43             case '/':
44                 op2 = pop();
45                 if (op2!=0.0)
46                     push(pop() / op2);
47                 else
48                     printf("error: zero divisor\n");
49                 break;
50             case '\n':
51                 printf("%.8g\n", pop());
52                 break;
53             default:
54                 printf("error: unknown command %s\n", s);
55                 break;
56         }
57     }
58
59     return 0;
60 }
61
62
63 void push(double f)
64 {
65     if (sp<MAXVAL)
66         val[sp++] = f;
67     else
68         printf("error: stack full, can't push %g\n", f);
69 }
70
71
72 double pop(void)
73 {
74     if (sp>0)
75         return val[--sp];
76     else {
77         printf("error: stack empty\n");
78         return 0.0;
79     }
80 }
81
82
83 int getop(char s[])
84 {
85     int i, c;
86
87     while ((s[0]=c=getch())==' ' || c=='\t');
88     s[1] = '\0';
89
90     if (!isdigit(c) && c!='.')
91         return c;
92
93     i = 0;
94     if (isdigit(c))
95         while (isdigit(s[++i]=c=getch()));
96     if (c=='.')
97         while (isdigit(s[++i]=c=getch()));
98     s[i] = '\0';
99     if (c!=EOF)
100        ungetch(c);
101     return NUMBER;
102 }
103
104
105 int getch(void)
106 {
107     return (bufp>0)?buf[--bufp]:getchar();
108 }
109
110
111 void ungetch(int c)
112 {
113     if (bufp>=BUFSIZE)
114         printf("ungetch: too many characters\n");
115     else
116         buf[bufp++] = c;
117 }
118

```