```c
#include <stdio.h>
#define N 10


void inputArray(int a[], int n)
{
    for (int i=0; i<n; i++)
        scanf("%d", &a[i]);
}


int sequenceSearch(int a[], int value, int n)
{
    int i, index=-1;

    for (i=0; i<n; i++)
        if (a[i]==value) {
            index = i;
            break;
        }

    return index;
}


int main()
{
    int arr[N], num, result;

    // printf("Input %d Numbers:\n", N);
    inputArray(arr, N);
    // putchar('\n');
    // printf("Input search number:\n");
    scanf("%d", &num);

    result = sequenceSearch(arr, num, N);
    if (result != -1)
        printf("num is arr[%d]: %d", result, arr[result]);
    else
        printf("num does not exist.");

    return 0;
}

```

```c
#include <stdio.h>

#define MaxNum  20
#define StrLen 100


void udf_getString(char s[], int maxlength)
{
    int  i=0;
    char c;

    while ((c=getchar())!='\n' && i<maxlength-1)
        s[i++] = c;
    s[i] = '\0';
}


int udf_strlen(char s[])
{
    int i;

    for (i=0; s[i]; i++);
    return i;
}


void udf_sort(char s[][StrLen], int num)
{
    int  i, j, k;
    char tmp[StrLen];

    for (i=0; i<num-1; i++) {
        k = i;
        for (j=i+1; j<num; j++)
            if (udf_strlen(s[k])>udf_strlen(s[j]))
                k = j;
        for (j=0; tmp [j]=s[i][j]; j++);
        for (j=0; s[i][j]=s[k][j]; j++);
        for (j=0; s[k][j]=tmp [j]; j++);
    }
}


void udf_print(char s[][StrLen], int num)
{
    for (int i=0; i<num; i++)
        puts(s[i]);
}


int main()
{
    char s[MaxNum][StrLen];
    int  i, num;

    scanf("%d", &num);
    getchar();
    for (i=0; i<num; i++)
        udf_getString(s[i], StrLen);

    udf_sort(s, num);
    udf_print(s, num);

    return 0;
}
```

```c
#include <stdio.h>

#define M 5
#define N 6


void merge(int a[], int m, int b[], int n, int c[])
{
    int i, j, x;

    for (i=0; i<m+n; i++) {
        x = (i<m)?a[i]:b[i-m];
        for (j=i; j>0; j--)
            if (c[j-1]>x)
                c[j] = c[j-1];
            else
                break;
        c[j] = x;
    }
}


int main()
{
    int a[M], b[N], c[M+N];
    int i;

    for (i=0; i<M; i++)
        scanf("%d", &a[i]);
    for (i=0; i<N; i++)
        scanf("%d", &b[i]);

    merge(a, M, b, N, c);

    for (i=0; i<M+N; i++)
        printf("%d ", c[i]);

    return 0;
}
```

```c
#include <stdio.h>
#define N 6

typedef struct date {
    int  year;
    int  month;
} DATE;

struct book {
    int  num;
    char title[20];
    DATE ptime;
};

void sortBook(struct book[], int);


int main()
{
    struct book lib[N];
    int i;

    for (i=0; i<N; i++)
        scanf("%d%s%d%d", &lib[i].num, lib[i].title, &lib[i].ptime.year, &lib[i].ptime.month);
    sortBook(lib, N);
    for (i=0; i<N; i++)
        printf("%d %s %d %d\n", lib[i].num, lib[i].title, lib[i].ptime.year, lib[i].ptime.month);

    return 0;
}


void sortBook(struct book lib[], int n)
{
    int i, j, k;
    struct book temp;

    for (i=0; i<n-1; i++) {
        k = i;
        for (j=i+1; j<n; j++)
            if (lib[k].ptime.year>lib[j].ptime.year)
                k = j;
            else if(lib[k].ptime.year==lib[j].ptime.year &&
                    lib[k].ptime.month>lib[j].ptime.month)
                k = j;
        temp   = lib[i];
        lib[i] = lib[k];
        lib[k] = temp;
    }
}

```

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

#define PI 3.141592654

struct point {
    double x;
    double y;
};


void translation(struct point pt[], double tl_x, double tl_y, int num)
{
    for (int i=0; i<num; i++) {
        pt[i].x += tl_x;
        pt[i].y += tl_y;
    }
}


void scale(struct point pt[], double s_x, double s_y, int num)
{
    for (int i=0; i<num; i++) {
        pt[i].x *= s_x;
        pt[i].y *= s_y;
    }
}


void rotation(struct point pt[], double angle, int num)
{
    double a[2][2];
    struct point temp;

    angle = angle * PI / 180;
    a[0][0] =  cos(angle);
    a[0][1] = -sin(angle);
    a[1][0] =  sin(angle);
    a[1][1] =  cos(angle);

    for (int i=0; i<num; i++) {
        temp.x = pt[i].x;
        temp.y = pt[i].y;
        pt[i].x = temp.x*a[0][0]+a[0][1]*temp.y;
        pt[i].y = temp.x*a[1][0]+a[1][1]*temp.y;
    }
}


int main()
{
    int i=0, num=0;
    char mode, action[10];
    double angle, tl_x, tl_y, s_x, s_y;
    struct point pt[10];

    do {
        printf("请输入坐标个数(>=2): ");
        scanf("%d", &num);
    } while (num<2);

    for (i=0; i<num; i++) {
        printf("请输入【第%d个】点的横x、纵y坐标: ", i+1);
        scanf("%lf%lf", &pt[i].x, &pt[i].y);
    }

    do {
        getchar();
        printf("请选择处理方式: 平移(t)、缩放(s)、旋转(r): ");
        mode = getchar();
    } while(mode!='t' && mode!='s' && mode!='r');

    switch (mode) {
        case 't':
            printf("请输入水平及垂直的平移量: ");
            scanf("%lf%lf", &tl_x, &tl_y);
            translation(pt, tl_x, tl_y, num);
            strcpy(action, "平移");
            break;

        case 's':
            printf("请输入水平及垂直的缩放比例: ");
            scanf("%lf%lf", &s_x, &s_y);
            scale(pt, s_x, s_y, num);
            strcpy(action, "缩放");
            break;

        case 'r':
            printf("请输入旋转角度: ");
            scanf("%lf", &angle);
            rotation(pt, angle, num);
            strcpy(action, "旋转");
            break;
    }

    printf("经过【%s】处理后, 坐标值如下: \n", action);
    for (i=0; i<num; i++)
        printf("%f %f\n", pt[i].x, pt[i].y);

    return 0;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>

#define MAXOP   100
#define NUMBER  '0'
#define MAXVAL  100
#define BUFSIZE 100

void   push(double);
double pop(void);
int    getop(char[]);
int    getch(void);
void   ungetch(int);

int    sp=0;
double val[MAXVAL];
int    bufp=0;
char   buf[BUFSIZE];


int main()
{
    int    type;
    double op2;
    char   s[MAXOP];

    while ((type=getop(s))!=EOF) {
        switch (type) {
            case NUMBER:
                push(atof(s));
                break;
            case '+':
                push(pop()+pop());
                break;
            case '*':
                push(pop()*pop());
                break;
            case '-':
                op2 = pop();
                push(pop()-op2);
                break;
            case '/':
                op2 = pop();
                if (op2!=0.0)
                    push(pop()/op2);
                else
                    printf("error: zero divisor\n");
                break;
            case '\n':
                printf("%.8g\n", pop());
                break;
            default:
                printf("error: unknown command %s\n", s);
                break;
        }
    }

    return 0;
}


void push(double f)
{
    if (sp<MAXVAL)
        val[sp++] = f;
    else
        printf("error: stack full, can\'t push %g\n", f);
}


double pop(void)
{
    if (sp>0)
        return val[--sp];
    else {
        printf("error: stack empty\n");
        return 0.0;
    }
}


int getop(char s[])
{
    int i, c;

    while ((s[0]=c=getch())==' ' || c=='\t');
    s[1] = '\0';

    if (!isdigit(c) && c!='.')
        return c;

    i = 0;
    if (isdigit(c))
        while (isdigit(s[++i]=c=getch()));
    if (c=='.')
        while (isdigit(s[++i]=c=getch()));
    s[i] = '\0';
    if (c!=EOF)
        ungetch(c);
    return NUMBER;
}


int getch(void)
{
    return (bufp>0)?buf[--bufp]:getchar();
}


void ungetch(int c)
{
    if (bufp>=BUFSIZE)
        printf("ungetch: too many characters\n");
    else
        buf[bufp++] = c;
}
```