CHAPTER 5. ARRAYS AND CHARACTERS

MẢNG MỘT CHIỀU

Khái niệm mảng

- Mảng là một số hữu hạn các phần tử có *cùng một kiểu giá trị* và có *chung một tên gọi*.
- Mỗi phần tử biểu diễn được một giá trị.
- Số phần tử của mảng phải được xác định ngay từ khi định nghĩa mảng (đối với mảng tĩnh).
- Mỗi phần tử của mảng được truy cập trực tiếp thông qua tên mảng cùng với chỉ số của mảng.
- Ngôn ngữ C/C++ cho phép làm việc với mảng một chiều, mảng hai chiều hoặc mảng nhiều chiều.

Khai báo mảng một chiều

< kiểu dữ liệu> tên mảng[const n];

Ví dụ để khai báo mảng một chiều có tên là a chứa 10 phần tử thuộc kiểu số nguyên ta thực hiện câu lệnh như sau:

int a[10];

Phần tử của mảng một chiều sẽ được truy xuất thông qua *tên mảng*[chỉ số], chẳng hạn để truy xuất phần tử thứ i của mảng một chiều có tên là a ta viết a[i].

Chỉ số của phần tử đầu tiên của mảng trong C/C++ mặc nhiên là 0, chẳng hạn trong khai báo trên mảng a gồm các phần tử a[0], a[1], ..., a[9].

Lưu ý: Trong một số tình huống, để dễ diễn đạt, chúng ta có thể sử dụng chỉ số đầu tiên của mảng là 1 hoặc là một giá trị khác mà chấp nhận lãng phí một số ô nhớ dành cho mảng.

Nhập các phần tử

```
void nhapmang(int a[], int &n)
{
    cin>>n;
    for (int i=0;i<n;i++)
        cin>>a[i];
}
```

Duyệt các phần tử

```
void duyetmang(int a[], int n)
{
    for (int i=0;i<n;i++)
        <Thực hiện thao tác trên phần tử a_i>
}
```

Sơ đồ tính tổng các phần tử của mảng

(thiết kế theo kiểu có giá trị trả về)

```
int tinhtong(int a[], int n)
{
    int tong=0;
    for (int i=0;i<n;i++)
    if (a[i] thỏa mãn điều kiện bài toán)
            tong =tong+a[i];
    return tong;
}
```

(thiết kế theo kiểu không giá trị trả về)

```
void tinhtong(int a[], int n)
{
    int tong=0;
    for (int i=0;i<n;i++)
            if (a[i] thỏa mãn điều kiện bài toán)
            tong =tong+a[i];
    cout<<tong;
}
```

Sơ đồ đếm số phần tử thỏa điều kiện

```
int demso(int a[], int n)
{
    int dem=0;
    for (int i=0;i<n;i++)
            if (a[i] thỏa mãn điều kiện bài toán)
            dem++;
    return dem;
}
```

Sơ đồ đặt cờ hiệu

```
int datcohieu(int a[], int n)
{
    for (int i=0;i<n;i++)
            if (a[i] thỏa mãn điều kiện bài toán)
                    return 0/1;
    return 1/0;
    }
```

Xóa phần tử tại vị trí *k* trong mảng

Để xóa một phần tử tại vị trí *k* của mảng, ta thực hiện như sau:

- Dịch chuyển các phần tử từ vị trí thứ *k* + 1 đến cuối mảng qua trái một vị trí.
- Sau khi dịch chuyển xong thì giảm số phần tử của mảng đi một đơn vị; nghĩa là cho *n* = *n*-1.
- Do *n* thay đổi sau khi thực hiện hàm nên *n* cần được truyền theo kiểu tham biến.

```
void xoaphantu(int a[], int &n, int k)
{
    for (int i=k;i<n-1;i++)
        a[i]=a[i+1];
    n--;
}
```

Chèn phần tử vào mảng

Chèn phần tử *x* vào tại vị trí thứ *k* của mảng ta thực hiện như sau:

- Dịch chuyển các phần tử từ vị trí thứ cuối mảng đến vị trí *k* + 1 qua phải một vị trí;
- tiếp theo chèn phần tử *x* vào vị trí *k* và cuối cùng là tăng *n* lên một đơn vị; nghĩa là *n* = *n*+1.
- Cũng như hàm xóa ở trên; biến *n* phải được truyền theo kiểu tham biến.

```
void chenphantu(int a[], int &n, int k, int x)
{
    for (int i=n;i>k;i--)
    a[i]=a[i-1];
    a[k]=x;
    n++;
}
```

Đặt lính canh

- Phần tử lính canh có thể hiểu là một vị trí nào đó của mảng, hoặc là một giá trị nào đó của mảng hoặc cũng có thể là một giá trị tùy theo từng yêu cầu cụ thể của từng bài toán,... Sau đó tiến hành duyệt các phần tử của mảng, nếu có phần tử nào thỏa điều kiện của bài toán thì tiến hành thay đổi vị trí lính canh (hoặc giá trị canh). Vị trí lính canh này chính là thông tin cần tìm.
- Tuy nhiên, tùy thông tin đầu vào của vấn đề bài toán mà xác định thông tin về phần tử lính canh cho phù hợp.

Một số ví dụ về sơ đồ đặt lính canh

- Tìm giá trị lớn nhất
- Tìm giá trị nhỏ nhất
- Tìm giá trị nguyên tố lớn nhất (nếu không có số nguyên tố nào thì trả về giá trị ?)
- Tìm giá trị nguyên tố nhỏ nhất (nếu không có số nguyên tố nào thì trả về giá trị ?)

Bài toán sắp xếp

Sắp xếp n phần tử của mảng 1 chiều theo thứ tự tang dần

```
void sapxep(int a[], int n)
{   for (int i=0; i<n-1;i++)
    for (int j=i+1;j<n ;j++)
    if (a[i]>a[j])
            hoanvi(a[i],a[j])
}
```

trong đó hàm hoán vị được viết như sau:

```
void hoanvi(int &a, int &b)
{
    int temp=a;
            a=b;
            b=temp;
}
```

Bài toán tìm kiếm

Cho dãy n số nguyên $a_0,a_2,...,a_{n-1}$ và một số nguyên x. Hãy tìm xem x có thuộc vào dãy số trên hay không. Nếu tìm được ở vị trí thứ i thì xuất kết quả là i, ngược lại nếu không tìm thấy thì xuất kết quả là –1 (chú ý dãy bắt đầu từ chỉ số 0, nếu dãy có nhiều số bằng x thì xuất vị trí i nhỏ nhất).

```
int timkiem (int a[], int n, int x)
{
 int i = 0;
 while ( i<n  && a[i]!=x)
 i++;
 if( i==n)
 return -1; //       tìm hết nhưng không có x
 return i; //        tìm thấy a[i] là phần tử có khóa x
}
```

Ví dụ 1.

Cho mảng một chiều a chứa n số nguyên. Hãy viết các hàm thực hiện các công việc sau:

a. Tìm số nguyên tố lớn nhất; nếu không có trả về giá trị 0.
b. Sắp xếp các số chẵn về đầu, các số lẻ về cuối

MẢNG HAI CHIỀU

Khai báo mảng hai chiều

- Cú pháp khai báo mảng hai chiều

  < kiểu dữ liệu> tên mảng[const m][const n];

- Ví dụ sau đây khai báo một mảng hai chiều a tên là a gồm 20 dòng, 50 cột, các phần tử là các số nguyên.

  int c[20][50];

Cấp phát bộ nhớ cho mảng

- Các phần tử trong mảng được cấp phát các ô nhớ kế tiếp nhau trong bộ nhớ. Kích thước của mảng bằng kích thước một phần tử nhân với số phần tử của mảng.
- Mảng hai chiều *m* dòng *n* cột được xem như là một bảng hình chữ nhật chứa *m***n* phần tử cùng kiểu dữ liệu.

Ví dụ về một mảng hai chiều ba dòng, bốn cột chứa các số nguyên.

	cột 0	cột 1	cột 2	cột 3
dòng 0	2	3	4	6
dòng 1	3	8	4	7
dòng 2	3	1	2	5

Truy nhập đến thành phần của mảng

- Biến mảng lưu trữ địa chỉ ô nhớ đầu tiên trong vùng nhớ được cấp phát. Ngôn ngữ C/C++ đánh chỉ số các phần tử trong mảng bắt đầu từ 0. Các phần tử của mảng được truy nhập thông qua Tên mảng và Chỉ số của phần tử của phần tử trong mảng.
- Mỗi phần tử của mảng được truy xuất thông qua tên mảng[chỉ số dòng][chỉ số cột], chẳng hạn để truy xuất phần tử tại dòng thứ *i* cột thứ *j* của mảng *a*, ta ghi là *a*[*i*][*j*].

Khởi tạo giá trị cho mảng

Các phần tử của mảng có thể được khởi tạo giá trị ngay khi khai báo

Chẳng hạn: int a[4] = {1,4,6,2};

//các phần tử ở các vị trí 0,1,2,3.

int b[2][3]={ {1,2,3}, {4,5,6} }; //dòng 0 và 1

Số lượng giá trị khởi tạo không được lớn hơn số lượng phần tử trong mảng. Nếu số lượng này nhỏ hơn, các phần tử còn lại được khởi tạo giá trị 0.

int a[3][4] = { {1}, {4,5} };

//a[0][0]=1,a[1][0]=4, a[1][1]=5 //các giá trị khác của mảng a đều bằng 0.

int a[3][4] = { };

//Tất cả đều mang giá trị 0

int b[] = {2, 4, 6, 8, 10, 12, 14, 16};

//các phần tử từ b[8] trở đi được gán số nguyên ngẫu nhiên

Nhập mảng

```
1. void nhapmang(int a[maxm][maxn], int &m, int &n)
2. {
3.       cout<<"Nhap vao so dong: ";cin>>m;
4.       cout<<"Nhap vao so cot : ";cin>>n;
5.       for (int i=0;i<m;i++)
6.       for (int j=0;j<n;j++)
7.                cin>>a[i][j];
}
```

Xuất mảng

```
1. void xuatmang(int a[maxm][maxn], int  m, int  n)
2. {
3.       for (int i=0;i<m;i++)
4.       {
5.             for (int j=0;j<n;j++)
6.             cout<<a[i][j]<<" ";
7.             cout<<endl;
8.       }
9. }
```

Sơ đồ tính tổng

```
1. void tinhtong(int a[maxm][maxn], int  m, int  n)
2. {
3.         int tong=0;
4.         for (int i=0;i<m;i++)
5.         for (int j=0;j<n;j++)
6.                 if (a[i] thỏa mãn điều kiện bài toán)
7.                         tong = tong +a[i][j];
8.         cout<<"Tong cac phan tu cua mang la : "<< tong;
9. }
```

Sơ đồ đếm số phần tử thỏa điều kiện

```
1.int demso (int a[maxm][maxn], int  m, int  n)
2.{   int dem=0;
3.    for (int i=0;i<m;i++)
4.    for (int j=0;j<n;j++)
5.          if (a[i] thỏa mãn điều kiện bài toán)
6.                dem++;
7.    return dem;
}
```

Sơ đồ đặt lính canh

Phần tử lính canh có thể hiểu là một vị trí nào đó của mảng, hoặc là một giá trị nào đó của mảng hoặc cũng có thể là một giá trị tùy theo từng yêu cầu cụ thể của bài toán,... Sau đó tiến hành duyệt các phần tử của mảng, nếu có phần tử nào thỏa điều kiện của bài toán thì tiến hành thay đổi vị trí lính canh (hoặc giá trị canh). Tùy thông tin đầu vào của vấn đề bài toán mà xác định thông tin về phần tử lính canh cho phù hợp.

Ví dụ 2.

Cho mảng hai chiều a có m dòng, n cột; các phần tử là các số nguyên. Hãy tìm số chính phương lớn nhất, nếu không tìm thấy trả về giá trị 0.

```
1.int chinhphuongln(int a[maxm][maxn], int  m, int  n)
2.{
3.    int cpmax=0;   // lính canh
4.    for (int i=0;i<m;i++)
5.    for (int j=0;j<n;j++)
6.    if ((sqrt(a[i][j])==int(sqrt(a[i][j]))) &&
  (a[i][j]>cpmax))
7.          cpmax=a[i][j];
8.    return cpmax;
}
```

Đặt cờ hiệu

Duyệt lần lượt từng phần tử của mảng, nếu có phần tử thỏa điều kiện thì dừng thuật toán và trả về một giá trị *k* nào đó; nếu không có phần tử nào thỏa điều kiện thì trả về giá trị *k'* nào đó; giá trị *k'* mang ý đảo nghĩa với giá trị *k*. Thường thì kỹ thuật cờ hiệu này nhằm trả lời cho câu hỏi: *có hay không ? đúng hay sai ? thuộc hay không thuộc ?....*

Khác với kỹ thuật lính canh là luôn phải duyệt hết các phần tử của mảng, kỹ thuật cờ hiệu chỉ cần tìm một phần tử thỏa điều kiện.

Ví dụ 3

Cho mảng hai chiều a có m dòng, n cột; các phần tử là các số nguyên. Hãy kiểm tra xem mảng có chứa dòng nào tăng dần hay không ? Nếu có trả về giá trị 1; nếu không có trả về 0.

```
1.int  kiemtradongtang(int  a[maxm][maxn],  int
  m, int  n)
2.{
3.   int flag;
4.   for (int i=0;i<m;i++)
5.   {
6.        flag=1;  // đặt cờ
7.        for (int j=0;j<n-1;j++)
8.        if (a[i][j]>a[i][j+1])
9.             flag=0;
10.       if (flag==1) return 1;
11.  }
12.  return 0;
13.}
```

Sắp xếp

Ví dụ 4

Cho mảng hai chiều a có m dòng, n cột; các phần tử là các số nguyên. Hãy thực hiện các công việc sau đây:

a. *Sắp xếp các phần tử tăng dần trên mỗi dòng (từ trái qua phải).*

b. *Sắp các phần tử tăng dần trên mỗi dòng (từ trái qua phải) và tăng dần trên mỗi cột (từ trên xuống dưới), khi đó mỗi phần tử của dòng dưới phải lớn hơn tất cả các phần tử của dòng trên.*

Câu a

```
1.void sapdong(int a[maxm][maxn], int  m, int  n)
2.{
3.   for (int i=0;i<m;i++)
4.   for (int j=0;j<n-1;j++)
5.   for (int k=j+1;k<n;k++)
6.        if (a[i][j]>a[i][k])
7.             hoanvi(a[i][j],a[i][k]);
8.}
```

Câu b

Cách 1: Dùng mảng phụ

Chuyển dữ liệu từ mảng hai chiều *a* sang mảng một chiều *b*, sau đó sắp xếp các phần tử tăng dần trên mảng một chiều *b* và cuối cùng chuyển dữ liệu từ mảng một chiều *b* qua lại mảng hai chiều *a* (cách này bạn đọc tự giải xem như bài tập).

Cách 2: Không dùng mảng phụ

```
1. void sapxep(int a[maxm][maxn], int  m, int  n)
2. {
3.       for (int i=0;i<m*n-1;i++)
4.       for (int j=i+1;j<m*n;j++)
5.             if(a[i/n][i%n]>a[j/n][j%n])
6.                   hoanvi(a[i/n][i%n],a[j/n][j%n]);
}
```

Xử lý dòng, cột

Ví dụ 5

Cho mảng hai chiều a có m dòng, n cột; các phần tử là các số nguyên.

a. *Hoán đổi nội dung hai dòng k, l của mảng a.*

b. *Dịch xuống xoay vòng các dòng một vị trí b (dòng 1 cuối cùng sẽ đưa lên vị trí dòng đầu tiên).*

```
1.void  hoanvidong(int  a[maxm][maxn],int
  m, int n,int k,int l)
2.{
3.  for (int i=0;i<n;i++)
4.      hoanvi(a[k][i],a[l][i]);
5.}
```

```
1.void  xoayvong(int  a[maxm][maxn],  int
 m, int  n)
2.{
3.      for (int i=m-1;i>0;i--)
4.      hoanvidong(a,m,n,i,i-1);
5.}
```

MẢNG KÝ TỰ

KÝ TỰ VÀ CHUỖI KÝ TỰ

- Ký tự được đặt giữa hai dấu nháy đơn
- Chuỗi ký tự được đặt trong dấu nháy kép
- Chuỗi ký tự là một dãy các ký tự liên tiếp kết thúc bằng ký tự NULL(‘\0’)

string.h

- char *gets(char * str);
- char *puts (char * str);
- unsigned *strlen(const char *str);
- char* strcpy(char *dest, const char * src);
- int strcmp(char *s1, char *s2);

- char * strrev(char *st);
- strlwr(char *s);
- strupr(char *s);
- strcat(char *s1, char *s2);
- strchr(char *s,char c);
- strrchr(char *s,char c);
- strstr(char *s1, char *s2);
- void flushall();

ctype.h

- void putchar (char c);
- void getchar(char c);
- int isalnum(int c);
- int isalpha(int c);
- int isdigit(int c);

- int islower (int c);
- isupper(int c);
- int isspace(int c);
- int tolower(int c);
- int toupper(int c);
- int toascii(int c);

Lưu ý

- Với chuỗi *s*, cú pháp *s* + *k* sẽ trả về chuỗi tính từ ký tự thứ *k* trở về cuối của chuỗi *s*.
- Với chuỗi *s*, cú pháp *s*[*k*]='\0' trả về chuỗi *s* trong dó *s* được kết thúc ở vị trí *k*.

NHẬP/XUẤT KÝ TỰ

- **Hàm getchar();//**Dùng để đọc một ký tự từ bàn phím và trả về ký tự đó.
- **Hàm getch();//**Có cùng chức năng như hàm getchar(), nhưng getch() nhập ký tự mà không hiện ra màn hình, trong khi getchar() nhập ký tự và ký tự đó có hiện ra màn hình.
- **Hàm putchar();// X**uất một ký tự ra màn hình.

Ví dụ 6.

- Nhập vào 2 chuỗi và sau đó xuất 2 chuỗi vừa nhập lên màn hình.

```
1.  #include<iostream.h>
2.  #include<string.h>
3.  #include<alloc.h>
4.  int main()
5.  {
6.   char *s1;
7.   s1=new char[1000];
8.   gets(s1);
9.   char *s2;
10.  s2=new char[1000];
11.  gets(s2);
12.  cout<<s1<<endl; // có thể sử dụng puts(s1);
13.  cout<<s2<<endl; // có thể sử dụng puts(s2);
14.  delete s1;
15.  delete s2;
16. }
```

Ví dụ 7.

Nhập từ bàn phím một chuỗi s; giả thiết thêm rằng chuỗi không có khoảng trắng dư thừa ở đầu và cuối chuỗi và giữa các từ có duy nhất một khoảng trắng. Hãy viết các hàm thực hiện các công việc sau (các công việc là độc lập với nhau).

a. Tách một từ bên phải (bên trái) của chuỗi s.
b. Đưa các ký tự đầu của mỗi từ thành chữ hoa, còn các ký tự khác thành chữ thường.
c. Đếm xem chuỗi s có bao nhiêu từ,
d. Đếm xem chuỗi s có bao nhiêu từ bắt đầu bằng ký tự nguyên âm (các ký tự a, u, i,o, e là các nguyên âm).

e. Đếm số ký tự của mỗi từ của chuỗi s.
f. Sắp xếp các ký tự của chuỗi s theo chiều tăng (theo mã ASCII), trong đó các ký tự khoảng trắng giữ nguyên vị trí.
g. Cho biết tần số xuất hiện của các chữ cái trong chuỗi s (không kể ký tự trống).
h. Đếm số lần xuất hiện chuỗi y trong chuỗi s.

```
18.  char *tachtutrai(char *s)
19.  {
20.       return strrev(strrchr(strrev(s),' ')+1);
21.  }
22.  char *tachtuphai(char *s)
23.  {
24.       return strrchr(s,' ')+1;
25.  }
```

```
26.  char *chuanhoatu(char *s)
27.  {
28.        strlwr(s);
29.        for (int i=0;i<strlen(s);i++)
30.              if (s[i]==' ' && s[i+1]!=' ' )
31.                    s[i+1]=toupper(s[i+1]);
32.        if (s[0]!=' ')
33.              s[0]=toupper(s[0]);
34.        return s;
35.  }
```

```
36.  int demsotu(char *s)
37.  {
38.        int l=strlen(s),d=1;
39.        for (int i=0;i<l;i++)
40.             if (s[i]==' ' && s[i+1]!=' ')
41.                  d++;
42.        return d;
43.  }
```

```
44.  int demtubatdaunguyenam(char s[])
45.  {
46.       int l=strlen(s),d=0;
47.       for (int i=0;i<l;i++)
48.            if (s[i]==' ' && (s[i+1]=='a' ||s[i+1]=='u' ||
  s[i+1]=='i' ||
49.                 s[i+1]=='o' || s[i+1]=='e'))
50.                      d++;
// Kiểm tra ký tự đầu tiên
51.       if (s[0]=='a' ||s[0]=='u' || s[0]=='i' || s[0]=='o'
  || s[0]=='e')
52.            d++;
53.       return d;
54.  }
```

```
55.  void demkytucuamoitu(char *s)
56.  {
57.        int sokytumoitu=0;
58.        for (int i=0;i<strlen(s);i++)
59.        if (s[i]!=' ') sokytumoitu++;
60.        else
61.        {
62.              cout<<sokytumoitu<<" ";
63.              sokytumoitu=0;
64.        }
65.              cout<<sokytumoitu;
66.  }
```

```
67.  void sapxepkytutang(char *s)
68.  {
69.        for(int i=0;i<strlen(s)-1;i++)
70.        for(int j=i+1;j<strlen(s);j++)
71.             if(*(s+i)>*(s+j)  &&  *(s+i)!=' '  &&*(s+j)!=' ' )
72.             {
73.                  char temp=*(s+i);
74.                  *(s+i)=*(s+j);
75.                  *(s+j)=temp;
76.             }
77.  }
```

```
78.  void tansocackytu( char *s)
79.  {
80.       int l=strlen(s);
81.       int d[255];
82.       for (int i=0;i<l;i++)
83.            d[s[i]]=0;
84.       for (i=0;i<l;i++)
85.       d[s[i]]++;
86.  for (i=0;i<l;i++)
87.       if (d[s[i]]!=0)
88.       {
89.            cout<<s[i]<<"  xuat  hien  "<<d[s[i]]<<"  lan"
  <<endl;
90.            d[s[i]]=0;
91.       }
92.  }
```

```
93.  int demchuoicon(char *s, char *y)
94.  {
95.        int d=0;
96.        while (strstr(s,y)!=NULL)
97.              {
98.              d=d+1;
99.              strcpy(s,strstr(s,y)+1);
100.       }
101. return d;
102. }
```

BT1.

Cho mảng n số nguyên dương (với $n \leq 30000$).

Hãy viết một chương trình hoàn chỉnh thực hiện các công việc sau đây (mỗi câu 1 điểm)

a.Đếm xem trong mảng có bao nhiêu cặp số nguyên tố cùng nhau ? (x,y) và (y,x) được tính là một cặp;

b.Giá trị nào trong mảng xuất hiện nhiều lần nhất; đếm số lần xuất hiện tương ứng.
Đếm xem trong mảng chứa bao nhiêu giá trị khác nhau ?

c.Đếm xem trong mảng có bao nhiêu số nguyên tố đối xứng ? (số nguyên tố đối xứng là một số nguyên tố bằng trung bình cộng của hai số nguyên tố liền trước và liền sau nó (lưu ý định nghĩa trong bài này không quan tâm đến việc 2 số nguyên tố kề với nó có thuộc dãy hay không)

d.Chuyển đổi mỗi số a_i về số nguyên tố nhỏ nhất lớn hơn hoặc bằng nó. Hãy tính tổng các số của mảng sau khi đã đổi.

e.Tìm một dãy con liên tiếp tăng dài nhất; xuất chiều dài của dãy tìm được.

Tìm một đoạn dài nhất gồm các phần tử liên tiếp trong dãy $\{a_i\}$: $a_l,a_{l+1},\ldots,a_r$ mà các phần tử đều là số nguyên tố; xuất chiều dài của dãy tìm được; nếu dãy không chứa số nguyên tố thì xuất giá trị 0.

f.Tìm cặp số nguyên tố liên tiếp (x,y) nhỏ hơn hoặc bằng n sao cho khoảng cách giữa x và y là lớn nhất. Xuất khoảng cách lớn nhất đó; khoảng cách bằng |x-y|; nếu không có tồn tại cặp số thỏa mãn xuất giá trị 0.

Ví dụ

16

1023 127 4000 12 7 29 3000 10 23 29 29 10 5 3000 31

9

Cau 1: 89

Cau 2: 3 12

Cau 3: 1

Cau 4: 11360

Cau 5: 4 3

Cau 6: 6

BT2

Cho dãy n số nguyên dương.

a. Tìm giá trị lớn nhất của dãy.

b. Đếm xem dãy có bao nhiêu số nguyên tố ?

c. Tìm tổng các chữ số của tất cả các số của dãy.

d. Tìm giá trị chẵn lớn nhất của dãy.

e. Tìm số nguyên tố lớn nhất của dãy.

BT3

Cho dãy n số nguyên dương.

a. Tìm các số nguyên tố trong dãy sao cho khi đảo ngược các chữ số của nó ta cũng thu được một số nguyên tố (ví dụ số 13, số 149,...).
b. Sắp xếp các số theo chiều tăng dần (<=).
c. Sắp xếp các số sao cho các số nguyên tố về cuối, các số còn lại về đầu dãy.
d. Tìm giá trị lớn thứ k của dãy.
e. Đếm số các đoạn con liên tiếp tăng (<=) của dãy a.
f. Tìm chiều dài của dãy con liên tiếp tăng (<=) dài nhất.

BT4.

Cho bảng số a gồm m dòng n cột, các phần tử là các số nguyên được nhập từ bàn phím.

a. Tìm tổng các phần tử của mảng.

b. Tìm giá trị nhỏ nhất của mảng.

c. Tính tổng các phần tử của mỗi dòng.

d. Tìm giá trị lớn nhất của mỗi dòng của mảng.

e. Đếm xem trong bảng có bao nhiêu số nguyên tố ?

f. Đếm xem trong mảng có bao nhiêu số nguyên tố đối xứng ?

BT5.

Cho bảng số a gồm n dòng n cột, các phần tử là các số nguyên được nhập từ bàn phím.

a. Tìm tổng các phần tử nằm trên đường chéo chính
b. Tìm tổng các phần tử nằm trên đường chéo phụ
c. Tìm tổng các phần tử nằm trong tam giác trên (trên đường chéo chính)
d. Tìm tổng các phần tử nằm trong tam giác dưới (dưới đường chéo chính)

BT6.

Cho bảng số a gồm m dòng n cột, các phần tử là các số nguyên được nhập từ bàn phím.

a. Hãy tạo mảng b có m dòng n cột biết rằng $b_{ij}=a_{ij} \times k_i$ với k_i là giá trị nhỏ nhất trên dòng i.
b. Hãy sắp xếp các phần tử tăng dần trên từng dòng
c. Hãy sắp xếp các phần tử tăng dần trên từng cột
d. Hãy sắp xếp các phần tử tăng dần từ trái qua phải và từ trên xuống dưới.

BT7

Cho mảng chứa các ký tự chữ hoa, chữ thường và ký tự khoảng trắng.

a. Đếm xem trong mảng có bao nhiêu ký tự chữ hoa ? Bao nhiêu ký tự là chữ thường.
b. Tìm tần số xuất hiện của các ký tự.
c. Tìm mã ASCII của mỗi ký tự.
d. Sắp xếp các ký tự theo chiều tăng (theo mã ASCII), các ký tự khoảng trắng giữ nguyên vị trí.
e. Các ký tự liền nhau gọi là một từ; hãy tìm một từ đầu tiên bên trái, một từ đầu tiên bên phải.
f. Tìm một từ dài nhất, từ dài nhất này có bao nhiêu ký tự ?
g. Hãy chuyển ký tự đầu của mỗi từ thành chữ hoa, các ký tự khác thành chữ thường.

BT8

Cho dãy n số nguyên $a_1,a_2,\ldots,a_n$ ($n \le 10^6$). Dãy con liên tiếp là dãy mà thành phần của nó là các thành phần liên tiếp nhau trong $\{a\}$, ta gọi tổng của dãy con là tổng tất cả các thành phần của nó. Tìm tổng lớn nhất trong tất cả các tổng của các dãy con của $\{a\}$

Ví dụ:

8

4 -5 6 -4 2 3 -7 6

Kết quả

7

BT9

Cho dãy gồm n số nguyên $a_1,a_2,\ldots,a_n$ ($n \le 1$ triệu) và số M. Đếm xem trong dãy có bao nhiêu cặp số có tổng bằng M. Xuất số lượng cặp số tìm được (hạn chế: Giả thiết các số trong dãy a_i đôi một khác nhau).

Ví dụ:

8 14

10 2 9 10 5 13 4 7

Kết quả

3

BT10

Nhập vào n số nguyên $a_1,a_2,\ldots,a_n$ (n<=30000,|a_i|<=10000). Số a_p (1<=p<=n) được gọi là một số trung bình cộng trong dãy nếu tồn tại 3 chỉ số i,j,k (1<=i,j,k<=n) đôi một khác nhau, sao cho $a_p=(a_i+a_j+a_k)/3$. Hãy tìm số lượng các số trung bình cộng trong dãy (p không nhất thiết phải khác các số i,j,k).

Ví dụ: Kết quả

5 1

2 9 5 7 10

BT11

Cho dãy n số nguyên dương ($n \le 10^6$; các số có giá trị nhỏ hơn 10^6). Hỏi dãy trên có bao nhiêu số có giá trị đôi một khác nhau ?

Ví dụ:

12

8 6 3 3 9 6 17 9 6 3 100 18

Kết quả

7

BT12

Cho bảng số A có *n* dòng n cột. Hãy tìm số nhỏ nhất trong số các số lớn nhất trên từng dòng của bảng số A.

BT13.

Cho bảng số A có *n* dòng n cột. Hãy tìm một bảng vuông con có kích thước lớn nhất chỉ chứa toàn số nguyên tố; xuất kích thước hình vuông con tìm được. Nếu bảng không có số nguyên tố nào xuất giá trị 0.

bảng A:

1023 127 4000 12

7 29 3000 10

23 29 29 10

5 3000 31 9

Kết quả:

Bài 12.

29

Bài 13.

2