///write a program that add two signal

#include <bits/stdc++.h>

using namespace std;

int main()

{

int n1,n2,x1[20],x2[20],y[20],root1,root2;

for(int i=1; i<20; i++)

{

x1[i] = 0;

x2[i] = 0;

}

cin>>n1;

for(int i=1; i<=n1; i++)

{

cin>>x1[i];

}

cin>>root1;

cin>>n2;

for(int i=1; i<=n2; i++)

{

cin>>x2[i];

}

cin>>root2;

int i,j;

for(i=root1,j=root2; i>=1 && j>=1; i--,j--)

{

y[max(i,j)] = x1[i]+x2[j];

}

while(i>=1)

{

y[i] = x1[i];

i--;

}

while(j>=1)

{

y[j]=x2[j];

j--;

}

for(int i=root1+1, j=root2+1; i<=max(n2,n1); j++,i++)

{

y[i] = x1[i]+x2[j];

}

for(int i=1; i<=max(n1,n2); i++)

cout<<y[i]<<" ";

return 0;

}

///write a program that multiply two signal

#include <bits/stdc++.h>

using namespace std;

int main()

{

int n1,n2,x1[20],x2[20],y[20],root1,root2;

for(int i=1; i<20; i++)

{

x1[i] = 0;

x2[i] = 0;

}

cin>>n1;

for(int i=1; i<=n1; i++)

{

cin>>x1[i];

}

cin>>root1;

cin>>n2;

for(int i=1; i<=n2; i++)

{

cin>>x2[i];

}

cin>>root2;

int i,j;

for(i=root1,j=root2; i>=1 && j>=1; i--,j--)

{

y[min(i,j)] = x1[i]\*x2[j];

}

for(int i=root1+1, j=root2+1; min(i,j)<=min(n2,n1); j++,i++)

{

y[min(i,j)] = x1[i]\*x2[j];

}

for(int i=1; i<=min(n1,n2); i++)

cout<<y[i]<<" ";

return 0;

}

//linear convolution

#include<bits/stdc++.h>

using namespace std;

int main()

{

int x\_size,k\_size,o\_x,o\_y,i,j,n;

cin>>x\_size>>k\_size>>o\_x>>o\_y;//jodi deya na thake tahle o\_x=o\_y=1 dite hobe

n=x\_size+k\_size-1;

int x[x\_size+1],k[k\_size+1];

memset(x,0,(x\_size+1)\*sizeof(x[0]));

memset(k,0,(k\_size+1)\*sizeof(k[0]));

for( i=1; i<=x\_size; i++)

{

cin>>x[i];

}

for( i=1; i<=k\_size; i++)

{

cin>>k[i];

}

vector<int>v;

for( i=k\_size; i>=o\_y; i--)

{

v.push\_back(k[i]);

}

for( i=o\_y-1; i>=1; i--)

{

v.push\_back(k[i]);

}

j=1;

int xy=1;

while(n--)

{

int ans=0;

bool b=true;

if(j<=x\_size)

{

b=false;

int xx=1;

for(i=j; i>=1; i--)

{

ans+=v[i-1]\*x[xx++];

}

j++;

}

else

{

int xx=x\_size;

int m=xy;

for(i=xx; i>=1 && m<k\_size; i--)

{

ans+=v[m++]\*x[xx--];

}

}

if(b)

xy++;

cout << ans << " ";

}

return 0;

}

//Deconvolution

#include<stdio.h>

int main()

{

int n,m,p,i,j;

int x[15],y[15],h[15];

///printf("Enter the sample size of output signal y(n) : ");

scanf("%d",&p);

scanf("%d",&m);

///printf("Enter the values of output signal y(n)\n");

for(i=0;i<p;i++)

scanf("%d",&y[i]);

///printf("Enter the sample size of impulse signal h(n) : ");

///printf("Enter the values of impulse signal h(n)\n");

for(i=0;i<m;i++)

scanf("%d",&h[i]);

n=p-m+1;

///append 0 to h(n)

for(i=m;i<n;i++)

h[i]=0;

///finding values for input signal x(n)

for(i=0;i<n;i++)

{

x[i]=y[i];

for(j=0;j<i;j++){

x[i]=x[i]-(x[j]\*h[i-j]);

}

x[i]=x[i]/h[0];

}

/// display input signal

///printf("The input signal is\n");

for(i=0;i<n;i++)

printf("%d ",x[i]);

}

///Concentric Circular Method:

#include<bits/stdc++.h>

using namespace std;

int main()

{

int x\_size,h\_size,i,j,ma;

cin>>x\_size>>h\_size;

ma=max(x\_size,h\_size);

int x[x\_size],h[h\_size],r[ma];

for(i=0; i<ma; i++)

{

x[i]=0;

h[i]=0;

}

for(i=0; i<x\_size; i++)

{

cin>>x[i];

}

for(i=0; i<h\_size; i++)

{

cin>>h[i];

}

for(i=0; i<ma; i++)

{

int n,ans=0;

for(j=0; j<ma; j++)

{

if((i-j)>=0)

n=i-j;

else

n=i-j+ma;

ans+=x[j]\*h[n];

}

cout << ans << " ";

}

return 0;

}

///Matrix Method to Calculate Circular Convolution

#include<bits/stdc++.h>

using namespace std;

int main()

{

int x\_size,h\_size,i,j,k=0;

cin>>x\_size;

int a[x\_size],h,ta[x\_size][x\_size],th[x\_size][1],ans[x\_size][1];

for(i=0; i<x\_size; i++)

cin>>a[i];

for(i=0; i<x\_size; i++)

{

cin>>h;

th[i][0]=h;

}

for(i=0; i<x\_size; i++)

{

for(j=0; j<x\_size; j++)

{

ta[j][i]=a[k++];

if(k==x\_size)

k=0;

}

k=x\_size-1-i;

}

for(i=0; i<x\_size; i++)

{

int sum=0;

for(j=0; j<x\_size; j++)

{

sum+=ta[i][j]\*th[j][0];

}

ans[i][0]=sum;

}

for(i=0; i<x\_size; i++)

{

cout << ans[i][0] << endl;

}

return 0;

}

///Even Components of a Signal

#include<bits/stdc++.h>

using namespace std;

int main()

{

int n, x[100],y[100],root;

cout<<"Enter a number(sapmle point)"<<endl;

cin>>n;

for(int i=1;i<=n;i++){

cin>>x[i];

}

cin>>root;

for(int i = 1; i<=root; i++)

{

y[i] = (x[i]+x[n-i+1])/2;

}

for(int i = root+1; i<=n; i++)

{

y[i] = (x[i]+x[n-i+1])/2;

}

for(int i=1; i<=n; i++)

{

cout<<y[i]<<" ";

}

return 0;

}

/// Odd Components of a Signal

#include<bits/stdc++.h>

using namespace std;

int main()

{

int n, x[100],y[100],root;

cout<<"Enter a number(sapmle point)"<<endl;

cin>>n;

for(int i=1;i<=n;i++){

cin>>x[i];

}

cin>>root;

for(int i = 1; i<=n; i++)

{

y[i] = (x[i]-x[n-i+1])/2;

}

for(int i=1; i<=n; i++)

{

cout<<y[i]<<" ";

}

return 0;

}

//continuous to discrete

#include <bits/stdc++.h>

#define pi acos(-1.0)

using namespace std;

int main()

{

int t;

double con[100],dis[100];

//continues signal to discrete signal

//5sin(PI\*t) for 0<=t<3 and time interval t = 0.1 (book page iss 7 example 2.1)

cin>>t;

int j = 0;

for(double i=0; i<=t; i=i+0.2,j++)

{

con[j]=5\*sin(pi\*i);

printf("%.3lf \n",con[j]);

}

j = 0;

for(double i=0; i<=t; i=i+0.2,j++)

{

dis[j]=5\*sin(pi\*j\*0.1);

printf("%.3lf %d\n",dis[j],j);

}

return 0;

}

///differentiation

#include<bits/stdc++.h>

using namespace std;

int main()

{

int n1,i,j;

cout<< "Enter Number of element of array = : "<<endl;

cin>>n1;

int x1[n1];

int t;

double re1[n1];

cout<<"Enter the array elements: "<<endl;

for(i=0;i<n1;i++)

{

cin>>x1[i];

}

cout<<"Enter the impulse signal = "<<endl;

cin>>t;

for(i=0;i<n1;i++)

{

re1[i]=(t-x1[i])\*1.0/t;

}

cout<<"The result: "<<endl;

for(i=0;i<n1;i++)

{

cout<<re1[i]<<" ";

}

return 0;

}