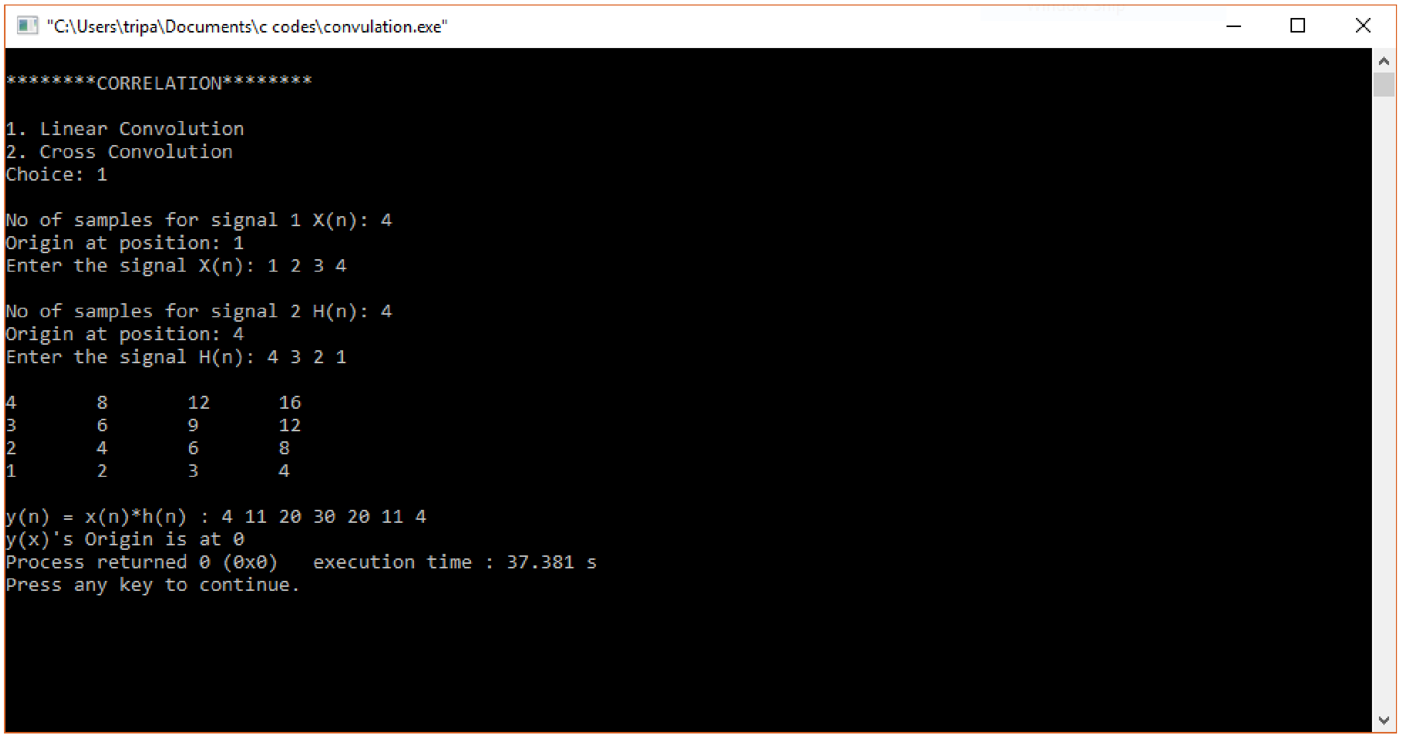
Code:
```
#include<stdio.h>
#include<math.h>
void computeCircularCorrelation(int [],int [],int [],int );
void computeLinearConvolution(int[],int [],int [],int ,int );
int main(void){
        int x[10],h[10],y[10];
        int i=0,n1=0,n2=0,j=0,l=0,k,m=0,n=0,o1=0,o2=0;
        int s[10];
        int matrix[10][10],col=0,row=0;
    int option=-1;

    printf("\n********CORRELATION********\n\n");
    printf("1. Linear Convolution\n");
    printf("2. Cross Convolution\n");
    printf("Choice: ");
    scanf("%d",&option);

    switch(option){
        case 1:
            printf("\nNo of samples for signal 1 X(n): ");
            scanf("%d",&n1);
            printf("Origin at position: ");
            scanf("%d",&o1);
            printf("Enter the signal X(n): ");

            for(i=0; i<n1; i++){
                scanf("%d",&x[i]);
            }

            printf("\nNo of samples for signal 2 H(n): ");
            scanf("%d",&n2);
            printf("Origin at position: ");
            scanf("%d",&o2);
            printf("Enter the signal H(n): ");

            for(i=0; i<n2; i++){
                scanf("%d",&h[i]);
            }

            l = n1+n2-1;

            col = n1;
            row = n2;
            computeLinearConvolution(x,h,s,col,row);
            printf("\ny(n) = x(n)*h(n) : ");
            for(i=0;i<l;i++)
                printf("%d ",s[i]);
    printf("\ny(x)'s Origin is at 0",o1+o2-1);

    break;

case 2:
    printf("\nNo of samples for signal 1 X(n): ");
    scanf("%d",&n1);
    printf("Enter the signal X(n): ");

    for(i=0; i<n1; i++){
        scanf("%d",&x[i]);
    }

    printf("\nNo of samples for signal 2 H(n): ");
    scanf("%d",&n2);
    printf("Enter the signal H(n): ");
    for(i=0; i<n2; i++){
        scanf("%d",&h[i]);
    }

    if(n1>n2){
        for(i=n2;i<n1;i++){
            h[i] = 0;
        }
        l = n1;
    }
    else if(n2>n1){
        for(i=n1;i<n2;i++){
            x[i] = 0;
        }
        l = n2;
    }

    printf("\nX(n): ");
    for(i=0; i<l; i++){
        printf("%d ",x[i]);
    }
    printf("\nH(n): ");
    for(i=0; i<l; i++){
        printf("%d ",h[i]);
    }
    computeCircularConvolution(x,h,s,l);
    printf("\ny(n) = x(n)*h(n) : ");
    for(i=0;i<l;i++)
        printf("%d ",s[i]);

    break;
```

```
        default :
            break;

    }
            return 0;
}
void computeCircularConvolution(int x[],int h[],int y[],int len){
    int i=0,j=0,c=0,sum=0;
    int matrix[len][len];
    for(i=0;i<len;i++){
        for(j=0;j<len;j++){
            matrix[j][i] = x[c%len];
            c++;
        }
        c = c - 1;
    }
    printf("\n\n");
    for(i=0;i<len;i++){
        for(j=0;j<len;j++){
            printf("%d ",matrix[i][j]);
        }
        printf("\n");
    }

    for(i=0;i<len;i++){
        for(j=0;j<len;j++){
            sum = sum + matrix[i][j]*h[j];
        }
        y[i] = sum;
        sum = 0;
    }


}
void computeLinearConvolution(int t[],int t_[],int s[],int col,int row){
    int i=0,j=0,k=0,m=0,mat[10][10];
    int l = col + row - 1;
    printf("\n");
    for(i=0;i<row;i++){
                    for(j=0;j<col;j++){
                            mat[i][j] = t[j]*t_[i];
                            printf("%d\t",mat[i][j]);
                    }
                    printf("\n");
            }
            for(k=0;k<l;k++){
                    s[k]=0;
                    if(k<(l+1)/2){
                            for (i=0,j=m ; i<row && j>=0 ; i++, j--) {
```

```
                        s[k]= s[k] + mat[i][j];
                }
                        m++;
                }
                else{
                        for (i=m-(col-1),j=col-1 ; i<row && j>=0 ; i++, j--) {
                        s[k]= s[k] + mat[i][j];
                }
                        m++;
                }
        }
}
```

Output:



"C:\Users\tripa\Documents\c codes\convulation.exe"

```
********CORRELATION********

1. Linear Convolution
2. Cross Convolution
Choice: 1

No of samples for signal 1 X(n): 4
Origin at position: 1
Enter the signal X(n): 1 2 3 4

No of samples for signal 2 H(n): 4
Origin at position: 4
Enter the signal H(n): 4 3 2 1

4       8       12      16
3       6       9       12
2       4       6       8
1       2       3       4

y(n) = x(n)*h(n) : 4 11 20 30 20 11 4
y(x)'s Origin is at 0
Process returned 0 (0x0)   execution time : 37.381 s
Press any key to continue.
```