DATA COMMUNICATION ASSIGNMENT

(2019BITE072 & 2019BITE040)

18 OCTOBER 2021

```
#include <GL/glut.h>
#include<iostream>
#include<sstream>
#include<cstring>
#include<windows.h>
#include<bits/stdc++.h>
#include<string>
#include<iostream>

using namespace std;
#define SIZE 100000 + 1
int a[100],n,x;

int P[SIZE * 2];
//Implementing Manacher Algorithm for liner O(n) time complexity to fing longest palindrome
subsequence
// Transform S into new string with special characters inserted.
string convertToNewString(const string &s) {
   string newString = "@";

   for (int i = 0; i < s.size(); i++) {
      newString += "#" + s.substr(i, 1);
   }

   newString += "#$";
   return newString;
}

string longestPalindromeSubstring(const string &s) {
   string Q = convertToNewString(s);
   int c = 0, r = 0;              // current center, right limit

   for (int i = 1; i < Q.size() - 1; i++) {
      // find the corresponding letter in the palidrome subString
      int iMirror = c - (i - c);
```

```
        if(r > i) {
            P[i] = min(r - i, P[iMirror]);
        }

        // expanding around center i
        while (Q[i + 1 + P[i]] == Q[i - 1 - P[i]]){
            P[i]++;
        }

        // Update c,r in case if the palindrome centered at i expands past r,
        if (i + P[i] > r) {
            c = i;              // next center = i
            r = i + P[i];
        }
    }

    // Find the longest palindrome length in p.

    int maxPalindrome = 0;
    int centerIndex = 0;

    for (int i = 1; i < Q.size() - 1; i++) {

        if (P[i] > maxPalindrome) {
            maxPalindrome = P[i];
            centerIndex = i;
        }
    }
    return s.substr( (centerIndex - 1 - maxPalindrome) / 2, maxPalindrome);
}

int cmp(int n)
{
            if (n==0) return 1;
            else return 0;
}
std::string NumberToString (int Number)
{
            stringstream ss; ss << Number;
            return ss.str();
}
```

```
void init2D(float r, float g, float b)
{
            glClearColor(r,g,b,0.0);
            glMatrixMode (GL_PROJECTION);
            gluOrtho2D (0.0, 200.0, 0.0, 150.0);
}
void printtext(int x, int y, string String)
{
//(x,y) is from the bottom left of the window
   glMatrixMode(GL_PROJECTION);
   glPushMatrix();
   glLoadIdentity();
   gluOrtho2D (0.0, 200.0, 0.0, 150.0);
   glMatrixMode(GL_MODELVIEW);
   glPushMatrix();
   glLoadIdentity();
   glPushAttrib(GL_DEPTH_TEST);
   glDisable(GL_DEPTH_TEST);
   glRasterPos2i(x,y);
   for (int i=0; i<String.size(); i++)
      glutBitmapCharacter(GLUT_BITMAP_9_BY_15, String[i]);
   glPopAttrib();
   glMatrixMode(GL_PROJECTION);
   glPopMatrix();
   glMatrixMode(GL_MODELVIEW);
   glPopMatrix();
}
void display()
{
            int b[50],c[100],t=0,z;
            for(int i=0;i<n;i++)
            b[i]=a[i];
            string s;
            glClear(GL_COLOR_BUFFER_BIT);
            for(int j=0;j<=1;j++)
            {
                        glColor3f(0.0, 0.0, 0.0);
                        s="0";printtext(16,99.8-60*j,s);
                        s="1";printtext(16,109.8-60*j,s);
                        s="-1";printtext(15.5,89.8-60*j,s);
                        glPushAttrib(GL_ENABLE_BIT);
                        glLineStipple(1,0xAAA0);
```

```
glEnable(GL_LINE_STIPPLE);
for(int i=0;i<n;i++)
{
glBegin(GL_LINES);
        glVertex2i(20+10*(i+1),80-60*j);
        glVertex2i(20+10*(i+1),120-60*j);
glEnd();
}
glPopAttrib();
glLineWidth(1.0);
glBegin(GL_LINES);
        glVertex2i(20,80-60*j);
        glVertex2i(20,120-60*j);
        glVertex2i(20,100-60*j);
        glVertex2i(10*(n+1)+30,100-60*j);
glEnd();
glPointSize(5.0);
glBegin(GL_POINTS);
        glVertex2i(20,110-60*j);
        glVertex2i(20,100-60*j);
        glVertex2i(20,90-60*j);
glEnd();
glLineWidth(2.0);
switch(x)
{
        case 1:         //NRZ-L
                if(j==1)
                for(int k=0;k<n;k++)
                a[k]=cmp(a[k]);
                for(int i=0;i<n;i++)
                {
                        glBegin(GL_LINES);
                        glVertex2i(20+10*i,100+20*a[i]-10-60*j);
                        glVertex2i(20+10*i+10,100+20*a[i]-10-60*j);
                        if(a[i]!=a[i+1]||i==n-1)
                        {
                                glVertex2i(20+10*i+10,100-10-60*j);
                                glVertex2i(20+10*i+10,100+20-10-60*j);
                        }
                        glEnd();
                        glBegin(GL_POINTS);
                        glVertex2i(20+10*(i+1),100-60*j);
                glEnd();
                s=NumberToString(i+1);
                 printtext(18+10*(i+1),96-60*j,s);
                 s=NumberToString(b[i]);
                 printtext(25+10*i,115-60*j,s);
        }
        break;
case 2:                 //NRZ-I
        if(j==1)
        {
                for(int i=0;i<n;i++)
                a[i]=b[i];
                a[0]=cmp(a[0]);
        }
        for(int i=0;i<n;i++)
        {
                if(i>0)
                if(a[i]==1)
                {
                        if(a[i-1]==0)
                        a[i]=1;
                        else a[i]=0;
                }
                else a[i]=a[i-1];
        }
        for(int i=0;i<n;i++)
        {
                glBegin(GL_LINES);
                glVertex2i(20+10*i,100+20*a[i]-10-60*j);
                glVertex2i(20+10*i+10,100+20*a[i]-10-60*j);
                if(a[i]!=a[i+1]||i==n-1)
                {
                        glVertex2i(20+10*i+10,100-10-60*j);
                        glVertex2i(20+10*i+10,100+20-10-60*j);
                }
                glEnd();
                glBegin(GL_POINTS);
                glVertex2i(20+10*(i+1),100-60*j);
                glEnd();
                s=NumberToString(i+1);
                 printtext(18+10*(i+1),96-60*j,s);
                 s=NumberToString(b[i]);
```

```
                    printtext(25+10*i,115-60*j,s);
            }
            break;
    case 3:                         //MANCHESTER
            if(j==0)
            {
                    for(int i=0;i<2*n;i++)
                    {
                            if(a[i]==0)
                            {
                                    c[t++]=1;
                                    c[t++]=0;
                            }
                            else
                            {
                                    c[t++]=0;
                                    c[t++]=1;
                            }
                    }
            }
            else
            {
                    for(int i=0;i<2*n;i++)
                    c[i]=cmp(c[i]);
            }
            for(int i=0;i<2*n;i++)
            {
                    glBegin(GL_LINES);
                    glVertex2i(20+5*i,100+20*c[i]-10-60*j);
                    glVertex2i(20+5*i+5,100+20*c[i]-10-60*j);
                    if(i==2*n-1||i%2==0)
                    {
                            glVertex2i(20+5*i+5,100-10-60*j);
                            glVertex2i(20+5*i+5,100+20-10-60*j);
                    }
                    if(i%2==0)
                    if(a[i/2]==a[i/2+1])
                    {
                            glVertex2i(20+5*i+10,100-10-60*j);
                            glVertex2i(20+5*i+10,100+20-10-60*j);
                    }
                    glEnd();
```

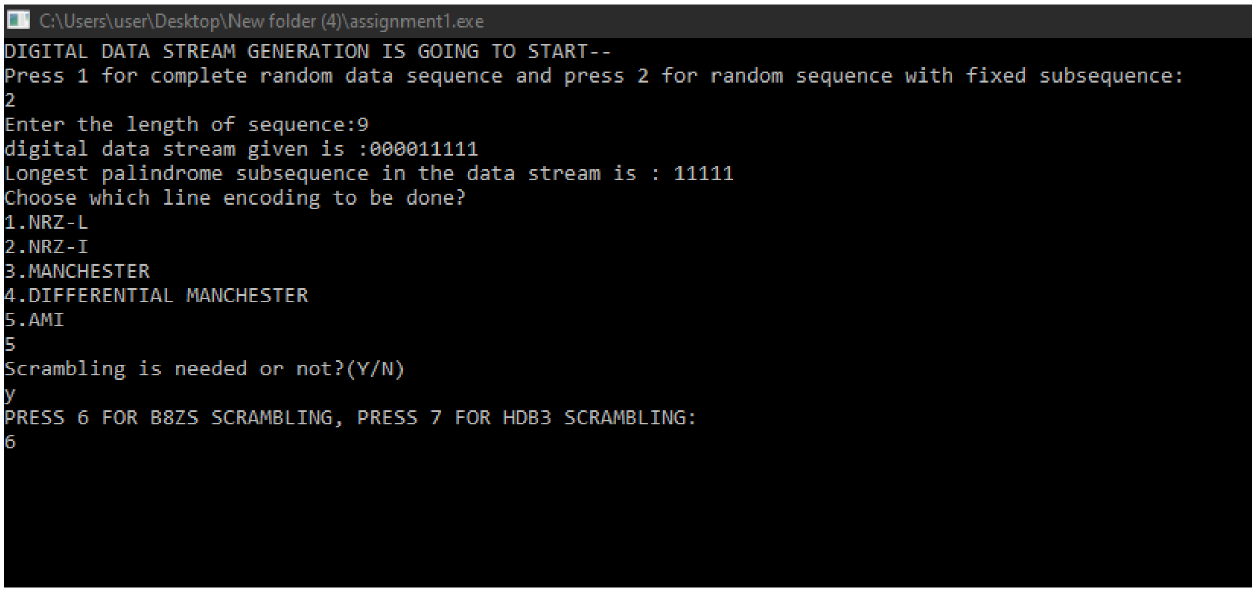
```
                        glBegin(GL_POINTS);
                        if(i%2!=0)
                        glVertex2i(20+5*(i+1),100-60*j);
                        glEnd();
                        if(i%2==0)
                        {
                                s=NumberToString(i/2+1);
                                printtext(18+10*(i/2+1),96-60*j,s);
                                 s=NumberToString(b[i/2]);
                                printtext(25+10*i/2,115-60*j,s);
                        }
                }
                break;
        case 4:                 //DIFFERENTIAL MANCHESTER
                t=0;
                if(j==1)
                {
                        for(int i=0;i<n;i++)
                        a[i]=b[i];
                        a[0]=cmp(a[0]);
                }
                for(int i=0;i<n;i++)
                {
                        if(i>0)
                        if(a[i]==1)
                        {
                                if(a[i-1]==0)
                                a[i]=1;
                                else a[i]=0;
                        }
                        else a[i]=a[i-1];
                }
                        for(int i=0;i<n;i++)
                        {
                                if(a[i]==0)
                                {
                                        c[t++]=1;
                                        c[t++]=0;
                                }
                                else
                                {
                                        c[t++]=0;
                        c[t++]=1;
                    }
                }
            for(int i=0;i<2*n;i++)
            {
                glBegin(GL_LINES);
                glVertex2i(20+5*i,100+20*c[i]-10-60*j);
                glVertex2i(20+5*i+5,100+20*c[i]-10-60*j);
                if(i==2*n-1||i%2==0)
                {
                    glVertex2i(20+5*i+5,100-10-60*j);
                    glVertex2i(20+5*i+5,100+20-10-60*j);
                }
                if(i%2==0)
                if(a[i/2]==a[i/2+1])
                {
                    glVertex2i(20+5*i+10,100-10-60*j);
                    glVertex2i(20+5*i+10,100+20-10-60*j);
                }
                glEnd();
                glBegin(GL_POINTS);
                if(i%2!=0)
                glVertex2i(20+5*(i+1),100-60*j);
                glEnd();
                if(i%2==0)
                {
                s=NumberToString(i/2+1);
                 printtext(18+10*(i/2+1),96-60*j,s);
                 s=NumberToString(b[i/2]);
                 printtext(25+10*i/2,115-60*j,s);
                }
            }
            break;
        case 6:                         //  SCRAMBLING
            z=-1;t=0;
            if(j==0)
            for(int i=0;i<n;i++)
            {
                if(a[i]==1)
                {
                    t++;
                    a[i]=-z;
```

```
                                        z=a[i];
                                }
                                else if(i+8<=n)
                                if(a[i]+a[i+1]+a[i+2]+a[i+3]+a[i+4]+a[i+5]+a[i+6]+a[i+7]==0)
                                {

s="V";printtext(25+10*(i+3),119,s);printtext(25+10*(i+3),59,s);

s="B";printtext(25+10*(i+4),119,s);printtext(25+10*(i+4),59,s);

s="V";printtext(25+10*(i+6),119,s);printtext(25+10*(i+6),59,s);

s="B";printtext(25+10*(i+7),119,s);printtext(25+10*(i+7),59,s);
                                        a[i+3]=z;a[i+4]=-z;a[i+6]=a[i+4];a[i+7]=a[i+3];
                                        z=a[i+7];
                                        i=i+7;
                                }
                        }
                        else
                                for(int i=0;i<n;i++)
                                        a[i]=-a[i];
                                goto case5;
                case 7: //       SCRAMBLING HDB3
                        z=-1;t=0;
                        if(j==0)
                        for(int i=0;i<n;i++)
                        {
                                if(a[i]==1)
                                {
                                        t++;
                                        a[i]=-z;
                                        z=a[i];
                                }
                                else if(i+4<=n)
                                if(a[i]+a[i+1]+a[i+2]+a[i+3]==0)
                                {
                                        if(t%2==0)
                                        {

s="B";printtext(25+10*(i),119,s);printtext(25+10*(i),59,s);

s="V";printtext(25+10*(i+3),119,s);printtext(25+10*(i+3),59,s);
```

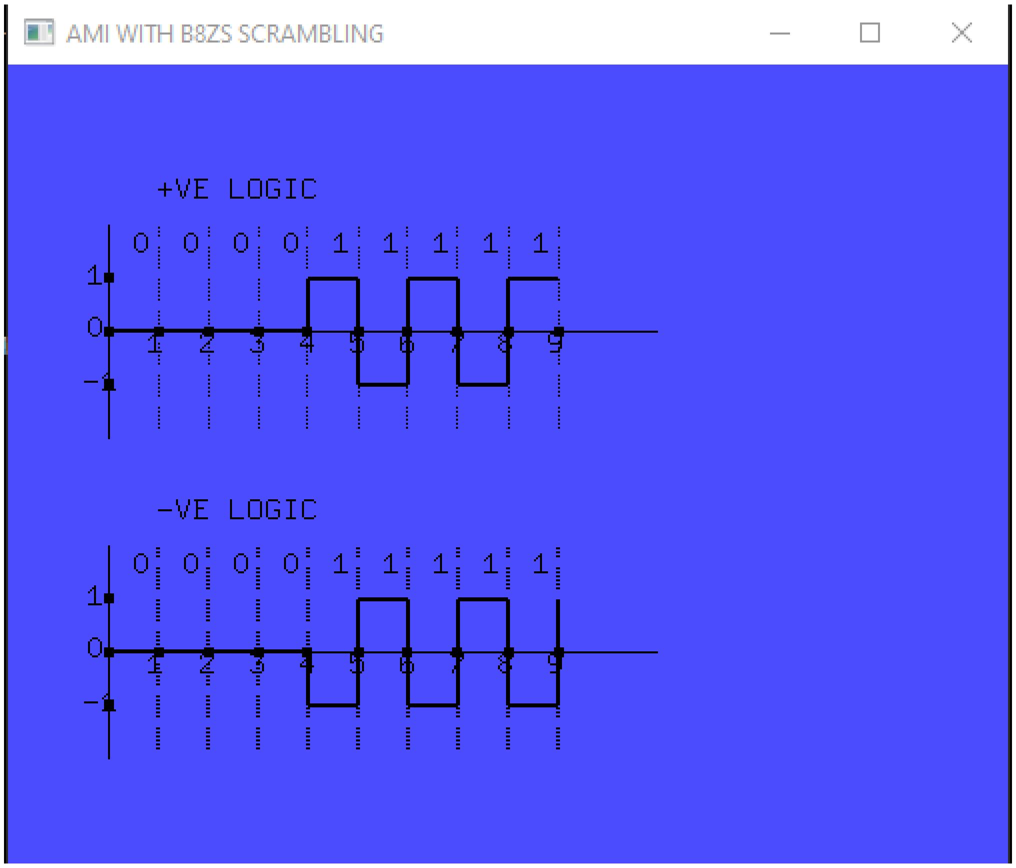
```
                                                        a[i]=-a[i-1];
                                                        a[i+3]=a[i];
                                                }
                                                else
                                                {

s="V";printtext(25+10*(i+3),119,s);printtext(25+10*(i+3),59,s);
                                                        a[i+3]=a[i-1];
                                                        t++;
                                                }
                                                z=a[i+3];
                                                i=i+3;
                                        }
                                }
                                else
                                        for(int i=0;i<n;i++)
                                                a[i]=-a[i];
                                        goto case5;
                        case 5:// AMI
                                t=0;
                                if(j==1)
                                {
                                        for(int i=0;i<n;i++)
                                        {
                                                a[i]=b[i];
                                                a[i]=cmp(a[i]);
                                        }
                                }
                                for(int i=0;i<n;i++)
                                {
                                        if(a[i]==1)
                                        {
                                                t++;
                                                if(t%2==0)
                                                a[i]=-1;
                                        }
                                }
                                case5:
                                for(int i=0;i<n;i++)
                                {
                                        glBegin(GL_LINES);
                                        glVertex2i(20+10*i,100+10*a[i]-60*j);
                                        glVertex2i(20+10*i+10,100+10*a[i]-60*j);
                                        if(a[i]!=a[i+1]||i==n-1)
                                        {
                                                glVertex2i(20+10*i+10,100+10*a[i]-60*j);
                                                glVertex2i(20+10*i+10,100+10*a[i+1]-60*j);
                                        }
                                        glEnd();
                                        glBegin(GL_POINTS);
                                        glVertex2i(20+10*(i+1),100-60*j);
                                        glEnd();
                                        s=NumberToString(i+1);
                                         printtext(18+10*(i+1),96-60*j,s);
                                         s=NumberToString(b[i]);
                                         printtext(25+10*i,115-60*j,s);
                                }
                                break;
                        }
                        if(x==5)
                        {
                                s="AMI";
                                if(j==1) s="Pseudoternary";
                        }
                        else
                        {
                                s="+VE LOGIC";
                                if(j==1) s="-VE LOGIC";
                        }
                        printtext(30,125-60*j,s);
                }
                glFlush();
                for(int i=0;i<n;i++)
                a[i]=b[i];
}
int main(int argc,char *argv[])
{
        char scramble;
        glutInit(&argc,argv);
        glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
        glutInitWindowSize (500, 400);
        glutInitWindowPosition (400, 100);
                int k,choice;
                cout<<"DIGITAL DATA STREAM GENERATION IS GOING TO START--"<<endl;
```

```
   cout<<"Press 1 for complete random data sequence and press 2 for random sequence with fixed
subsequence:"<<endl;
   cin>>choice;
   cout<<"Enter the length of sequence:";
   cin>>n;
   srand(time(0));
   if(choice==1)
   {
   for(int i=0;i<n;i++)
   a[i]=rand()%2;
   }
   if(choice==2)
   {
   for(int i=0;i<4*n;i=i+4)
   {
   k=rand()%2;
     a[i]=k;
   a[i+1]=k;
   a[i+2]=k;
   a[i+3]=k;
  }
  }
  cout<<"digital data stream given is :";
  for(int i=0;i<n;i++)
  {
 cout<<a[i];
  }
  cout<<endl;
  string str="";

for(int i=0;i<n; i++)
  str+= (a[i]+48);

cout<<"Longest palindrome subsequence in the data stream is : " <<longestPalindromeSubstring(str);
//implemented manacheralgorithm for linear time complexity

                printf("\nChoose which line encoding to be done?\n");
                printf("1.NRZ-L\n2.NRZ-I\n3.MANCHESTER\n4.DIFFERENTIAL MANCHESTER\n");
                printf("5.AMI \n");
                cin>>x;
                if(x==5){
                printf("Scrambling is needed or not?(Y/N)\n");
		cin>>scramble;
		if(scramble=='Y'|| scramble=='y')
		{
		cout<<"PRESS 6 FOR B8ZS SCRAMBLING, PRESS 7 FOR HDB3 SCRAMBLING: \n";
		cin>>x;
	}}
		switch(x)
		{
			case 1:
				glutCreateWindow ("NRZ-L ENCODING");
				break;
			case 2:
				glutCreateWindow ("NRZ-I ENCODING");
				break;
			case 3:
				glutCreateWindow ("MANCHESTER ENCODING");
				break;
			case 4:
				glutCreateWindow ("DIFFERENTIAL MANCHESTER ENCODING");
				break;
			case 5:
				glutCreateWindow ("AMI AND PSEODOTERNARY ENCODING");
				break;
			case 6:
				glutCreateWindow ("AMI WITH B8ZS SCRAMBLING");
				break;
			case 7:
				glutCreateWindow ("AMI WITH HDB3 SCRAMBLING");
				break;
		}
		init2D(0.3,0.3,2);
		glutDisplayFunc(display);
		glutMainLoop();
}
```

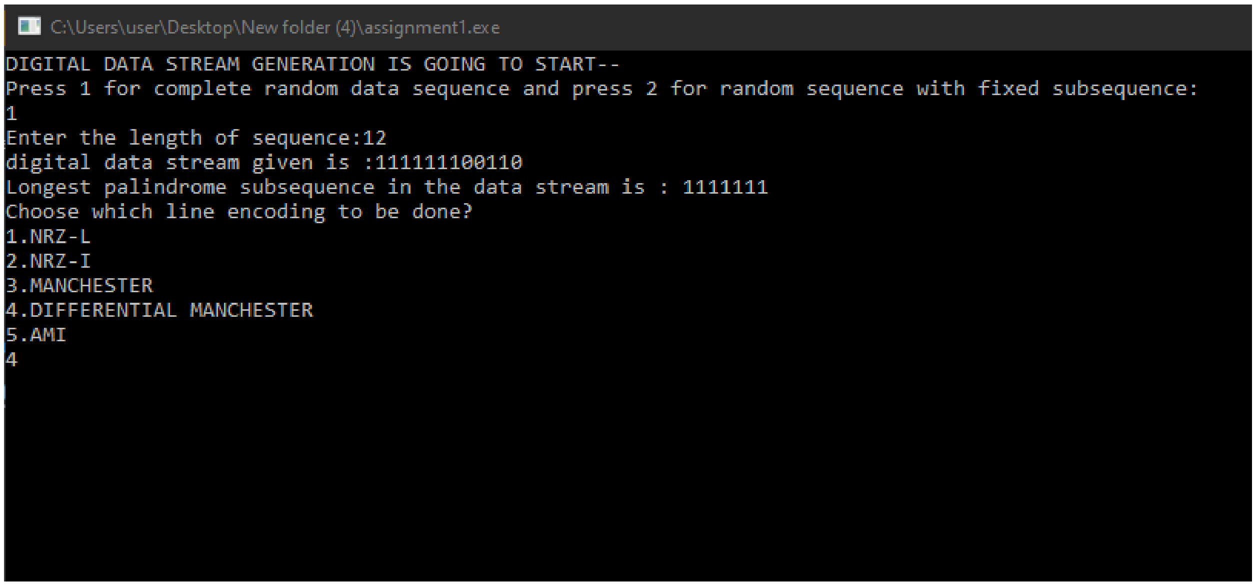
OUTPUT:



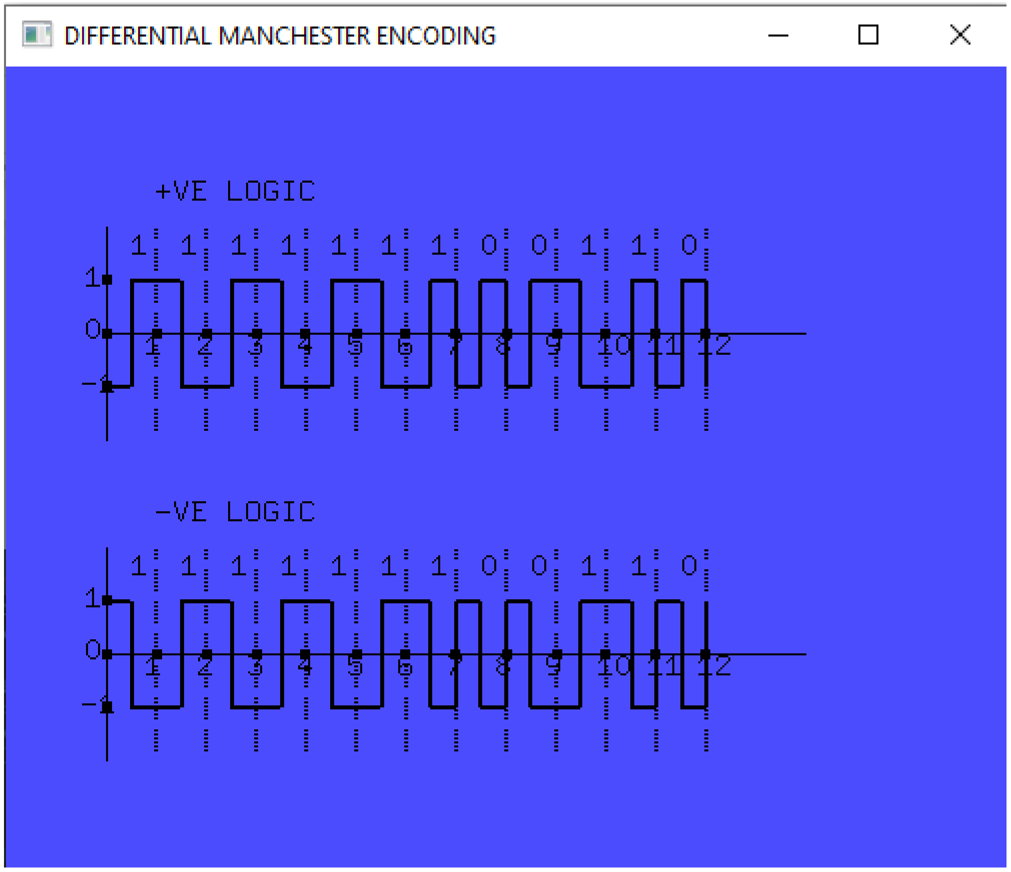
```
C:\Users\user\Desktop\New folder (4)\assignment1.exe
DIGITAL DATA STREAM GENERATION IS GOING TO START--
Press 1 for complete random data sequence and press 2 for random sequence with fixed subsequence:
2
Enter the length of sequence:9
digital data stream given is :000011111
Longest palindrome subsequence in the data stream is : 11111
Choose which line encoding to be done?
1.NRZ-L
2.NRZ-I
3.MANCHESTER
4.DIFFERENTIAL MANCHESTER
5.AMI
5
Scrambling is needed or not?(Y/N)
y
PRESS 6 FOR B8ZS SCRAMBLING, PRESS 7 FOR HDB3 SCRAMBLING:
6
```



OUTPUT:



```
C:\Users\user\Desktop\New folder (4)\assignment1.exe
DIGITAL DATA STREAM GENERATION IS GOING TO START--
Press 1 for complete random data sequence and press 2 for random sequence with fixed subsequence:
1
Enter the length of sequence:12
digital data stream given is :111111100110
Longest palindrome subsequence in the data stream is : 1111111
Choose which line encoding to be done?
1.NRZ-L
2.NRZ-I
3.MANCHESTER
4.DIFFERENTIAL MANCHESTER
5.AMI
4
```


DIFFERENTIAL MANCHESTER ENCODING
+VE LOGIC
1 1 1 1 1 1 1 0 0 1 1 0
1
0
-1
1 2 3 4 5 6 7 8 9 10 11 12
-VE LOGIC
1 1 1 1 1 1 1 0 0 1 1 0
1
0
-1
1 2 3 4 5 6 7 8 9 10 11 12