

Primer bloque de programas

Eslí Joana Osorio Rodríguez

12 de septiembre de 2016

Índice

1. Introducción	3
2. Universo	4
2.1. Descripción:	4
2.2. Código fuente	4
2.3. Pruebas	8
3. Números primos	9
3.1. Descripción	9
3.2. Código	9
3.3. Pruebas	13
4. Autómata terminación -ere	19
4.1. Descripción	19
4.2. Código	19
4.3. Pruebas	24
5. Autómata de paridad	27
5.1. Descripción	27
5.2. Código	27
5.3. Pruebas	31
6. Protocolo	34
6.1. Descripción	34
6.2. Código	34
6.3. Pruebas	37
7. Cadenas que terminan en 01	41

1. Introducción

Este documento contiene la documentación de todos los programas realizados en el primer parcial. Se dejaron seis programas en total, los resultados obtenidos y los códigos se encuentran a continuación. Los códigos se encuentran en dos lenguajes C y Python, estos lenguajes los escogí debido a que C es un lenguaje que ya sabía y Python por su facilidad aunque este ultimo lo tuve que aprender. La forma de programar autómatas es n poco diferente debido a que estos nos dan un resultado analizando estados, no con contadores u otro tipo de herramientas, al principio puede resultar difícil abrir la mente y dejar a un lado la forma en la cual programabas pero después se torna mucho más sencillo y te das cuenta de que muchas veces implementabas cosas que realmente no eran necesarias.

2. Universo

Un alfabeto básicamente es un conjunto finito de símbolos y es no vacío. Una cadena es la concatenación de símbolos pertenecientes al alfabeto. La cadena vacía si existe, no está vacía, pero no tiene símbolos. [?]

2.1. Descripción:

Crear un programa que basado en un alfabeto binario $\Sigma = \{0, 1\}$, así el programa debe de ser capaz de mostrarnos todas las combinaciones posibles que se puedan formar con base a el alfabeto el programa está limitado a las potencias $0 \leq n \leq 1000$.

2.2. Código fuente

El programa para está problema fue escrito en el lenguaje C

Archivo: main.c

```
#include "Random.h"
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>

void agregarCaracter(int,int,int,int,char []);
int comenzar(int);
void menu(void);
void Manual(void);
void Automatico(void);
void repetitivo(void);

int main()
{
    menu();
    repetitivo();
}

void menu(void) {
    system("color_8F");
    int rep;
    int opc=0;
    do{
        printf("\t\tUniverso_binario_1\n_Seleccione_la_opcion_deseada:\n_1.-
Manual\n2..-Automatico\n");
        scanf("%d",&opc);
```

```

        //opc=Random(1,2);
        if(opc==1){
            Manual();
        }
        if(opc==2){
            Automatico();
        }
        printf("\n\t\tDesea_ingresar_otro_valor?_\\n_1.-Si_\\n_2.-_No_\\n
            ");
        scanf("%d",&rep);
        //rep=Random(0,2)
    }while(rep==1);
    printf("\t\tAdios\\n");
}

void repetitivo(void){
    int num=0;
    printf("\nQuieres_que_se_repita?_\\n_1.Si_\\n_2._No_\\n");
    num=Random(1,2);
    if(num==1){
        printf("\n_Opcion_1_seleccionada\\n");
        menu();
    }else{
        printf("\n_Opcion_2_seleccionada,_Adios\\n");
        return ;
    }
}

void Manual(void){
    int potencia=0;
    printf("\t\tSelecciono_el_modos_Manual\\n");
    printf("Incerte_la_potencia_");
    scanf("%d",&potencia);
    printf("\nLa_potencia_insertada_es:_%d",potencia);
    comenzar(potencia);
}

void Automatico(void){
    int potencia;
    printf("\t\tSelecciono_el_modos_Automatico\\n");
    potencia=Random(0,1000);
    printf("La_potencia_seleccionada_automaticamente=_%d",potencia
    );
    comenzar(potencia);
}

int comenzar(int potencia){
    FILE *archivo;

```

```

char cadena[potencia];
int senal;
senal=potencia;
int cantidad=pow(2,potencia);
int contador=1;
archivo=fopen("archivo.txt","w");
fprintf(archivo,"A={");
for(int i=0;i<cantidad;i++){
    int posicion=0;

    do{
        if(potencia==0){
            return 0 ;
        }
        else{
            if(potencia==1 && contador%2==1){
                cadena[posicion]='0';
            }
            else{
                if(potencia==1 && contador%2==0){
                    cadena[posicion]='1';
                }
                else{
                    if(contador<=cantidad
                        /2){
                        cadena[
                            posicion]='
                            0';
                    }
                    else{
                        cadena[
                            posicion]='
                            1';
                        contador=
                            contador-(
                                cantidad/2)
                            ;
                    }
                }
            }
        }

        fprintf(archivo,"%e" ,cadena[posicion]);
        posicion++;
        potencia--;
    } while(posicion!=senal);

    for(int x=0;x<posicion;x++){
        fprintf(archivo,"%e" ,cadena[x]);
    }
}

```

```

    }

    if(i!=cantidad-1 && i<cantidad){
        fprintf(archivo,",");
    }
    cantidad++;

}
fprintf(archivo,"}");
fclose(archivo);
return 0;
}

```

2.3. Pruebas

En cuanto a las pruebas, a continuación se mostraran una serie de imágenes capturadas al momento de ejecutar el programa. Los resultados arrojados por el programa anterior son:

Este programa un no esta del todo bien no gurda bien las cadenas que genera

Para la selección en modo automático:

Figura 1: Selección de un $n = 24$ de forma automática

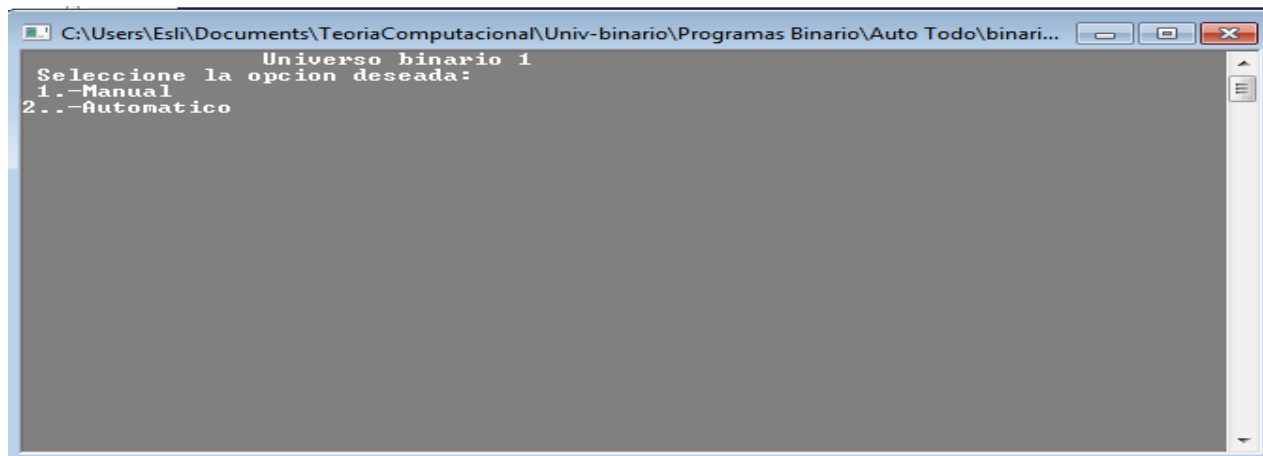


Figura 2: Consola

3. Números primos

El conjunto de números primos es infinito, pero números primos solo son aquellos que son divisibles entre ellos mismos y el número uno, para encontrar los números primos no existe ningún algoritmo estable, los números primos no llevan una secuencia ni un patrón por esto hasta la fecha no se existe un algoritmo para saber cuáles son o no primos.

3.1. Descripción

Crear un programa que encuentre los números primos dentro del intervalo: $0 \leq n \leq 1000$. Los números primos obtenidos deberán ser guardados en un archivo que contendrá el conjunto decimal y el conjunto binario, después se grafican por el número de ceros y unos de cada número primo en su representación binaria.

3.2. Código

El programa está escrito en C.

Archivo: main.py

```
#include "Random.h"
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>

int c=0;
int b=0;

void Conversion(int [],int );
int archivarDec(int , int );
void archivarBin(char Numero[] , int , int );
void Primos(int );
void Menu();
void Manual();
void Automatico();
int inicializarConjunto(int []);

int main(int argc, char** argv) {
    system ("color_8F" );
    FILE *fp;
    fp=fopen("archivo.txt","w");
    int rep;
    //Manual();

    do{
```

```

        rep=0;
        Menu();
        printf ("\t\tDesea_ingresar_otro_valor?\n_1=_Si_\n_2=No\n");
//        scanf("%d",&rep);
        rep=Random(0,2);

    }while(rep==1);

    printf ("\t\t_Gracias_Por_su_Visita_\n\t\t_Adios");
}

void Menu(void) {
    int opc=0;
    printf ("\t\t\t\tNUMEROS_PRIMOS\t\t\n_MENU_:\n_1._Operacion_Manual_\n_
2._Operacion_automatica\n\n");
//    scanf("%d",&opc);
    opc=Random(0,2);
    if(opc+1==1){
        Manual();
    }
    if(opc+1==2){
        Automatico();
    }
}

void Manual() {
    int limite=0;
    printf ("Usted_selecciono_la_forma_manual\n");
    printf ("introduzca_el_numero_de_limite=___");
    scanf(" %d",&limite);
    printf ("\n");
    Primos(limite);
}

void Automatico() {
    printf ("Usted_selecciono_la_forma_automatica\n");
    int limite=0;
    limite=Random(1,750);
    printf ("El_numero_random_del_limite_es=___%d\n", limite);
    Primos(limite);
}

void Coma() {
    FILE *fp;
    fp=fopen("archivo.txt","a");
    fprintf(fp,"");
}

void Cerrar() {

```

```

        FILE *fp;
        b=0;
        fp=fopen("archivo.txt","a");
        fprintf(fp,"}") ;
        fprintf(fp,"\n") ;
    }

    int archivarDec(int num, int b){
        int x[130];
        int i;
        x[i]=num;
        FILE *fp;
        if (b==0){
            fp=fopen("archivo.txt","a");
            fprintf(fp,"A={");
            fprintf(fp,"%d",x[i]);
        }
        if (b>0){
            fprintf(fp,"%d",x[i]);
        }

        i++;
        return 0;
    }

    void archivarBin(char Numero[15], int lim, int ban){
        int bit;
        int a;
        a=lim;
        FILE *fp;
        fp=fopen("archivo.txt","a");
        if (ban ==0)fprintf(fp,"Ab={");

        for (a=lim-1; a>=0;a--){
            fprintf(fp,"%d",Numero[a]);
        }
    }

    int inicializarConjunto(int Conjunto[171]){
        int i;
        for (i=0; i<=127; i++){
            Conjunto [i]=0;
        }
    }

    void Primos(int limite){
        int num;
        int f=0;
        int conjunto[171];
        int i=0;
        int b=0;
        int c=0;
    }

```

```

int a;
int bandera;
inicializarConjunto (conjunto);
        if (limite==0){
            FILE *fp;
            fp=fopen("archivo.txt","a");
            fprintf(fp,"A={e}");
            fprintf(fp,"\n");
            fprintf(fp,"Ab={}");
            fprintf(fp,"\n");
        }
for (num=1; num<=limite; num++){
    bandera=0;
    for(a=2; a<num; a++){
        if (num%a==0){
            bandera=1;
        }
    }
    if (bandera==0){
        conjunto [i]=num;
        if (f!=0){
            Coma();
        }
        archivarDec(num,b);
        c=1;
        b=1;
        f=1;
        i++;
    }

    if (num==limite) {
        Cerrar ();
        Conversion(conjunto,i);
    }
}
}

```

```

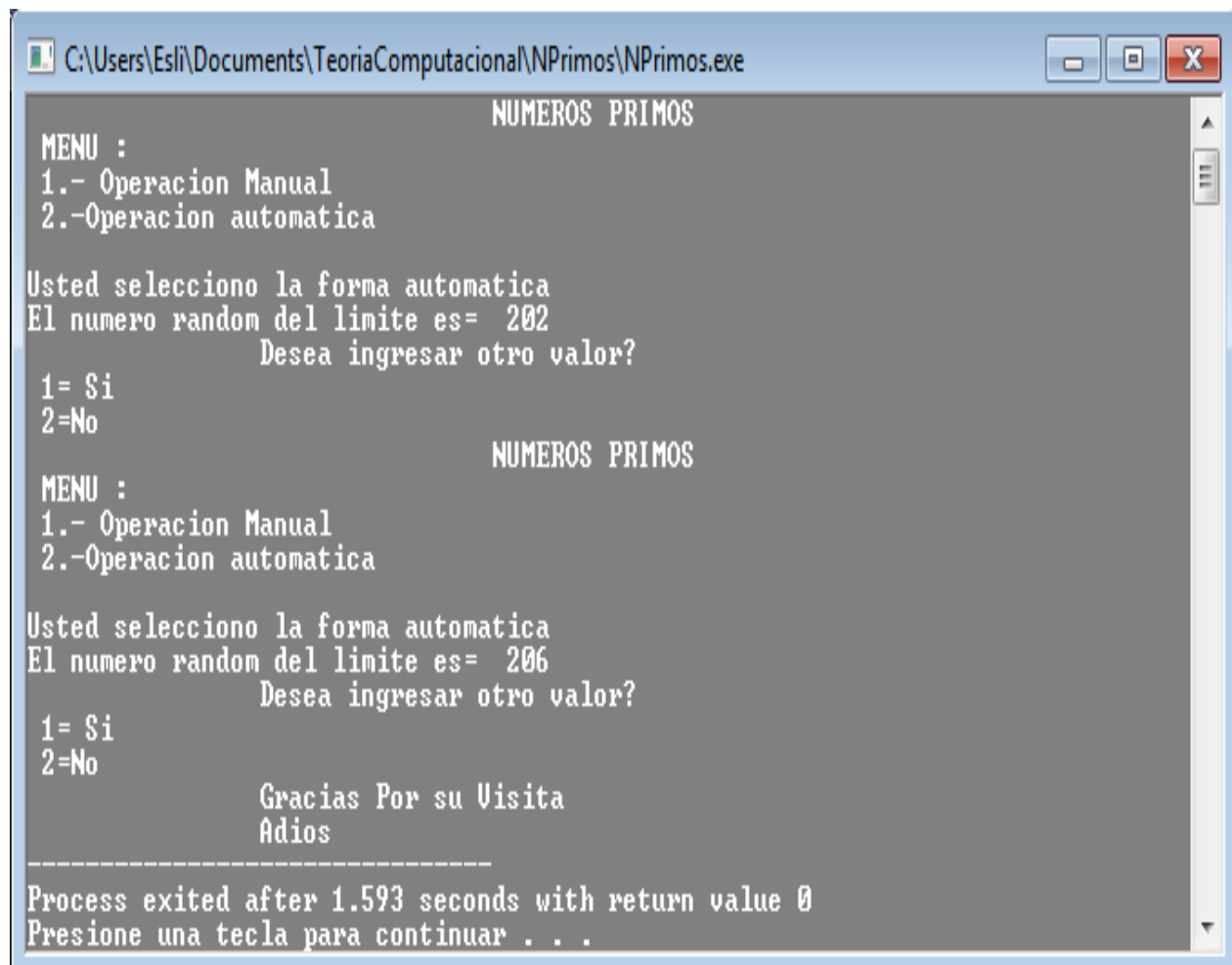
void Conversion (int conjunto [171], int lim){
    char numero [10];
    int bit;
    int decimal;
    int i=0;
    int h=0;
    int car;
    int ban=0;
do{
    decimal=conjunto[i];
    h=0;
    do{
        bit= decimal%2;
        numero[h]= (char) bit;
        decimal=decimal/2;
    }
}

```

```
        h++;
    } while (decimal>0);
        i++;
        archivarBin (numero,h,ban);
            if (i!=lim) Coma();
        ban=1;
} while (i!=lim);
    Cerrar();
    return ;
}
```

3.3. Pruebas

Imagenes del Programa ejecutandose.



```
NUMEROS PRIMOS
MENU :
1.- Operacion Manual
2.-Operacion automatica
Usted selecciono la forma automatica
El numero random del limite es= 202
      Desea ingresar otro valor?
1= Si
2=No
NUMEROS PRIMOS
MENU :
1.- Operacion Manual
2.-Operacion automatica
Usted selecciono la forma automatica
El numero random del limite es= 206
      Desea ingresar otro valor?
1= Si
2=No
      Gracias Por su Visita
      Adios
-----
Process exited after 1.593 seconds with return value 0
Presione una tecla para continuar . . .
```

Figura 3: Opción automática

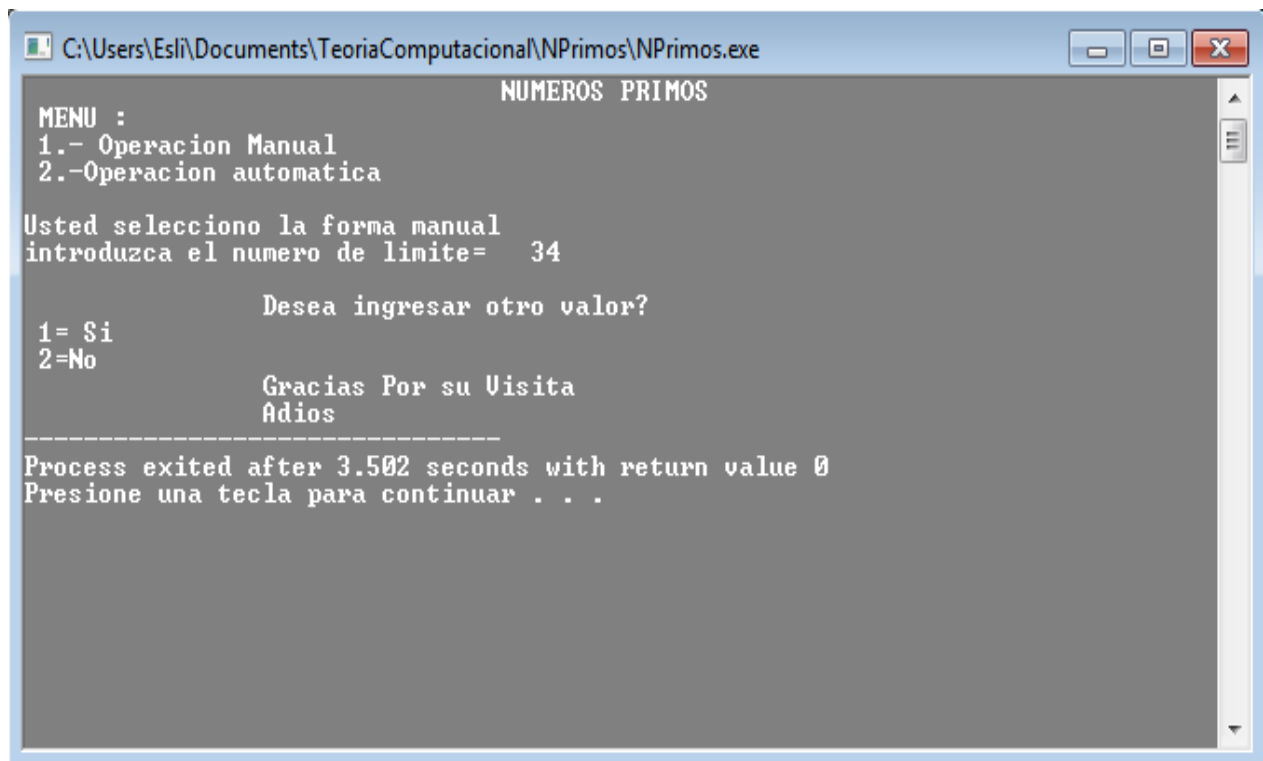


Figura 4: Opción manual

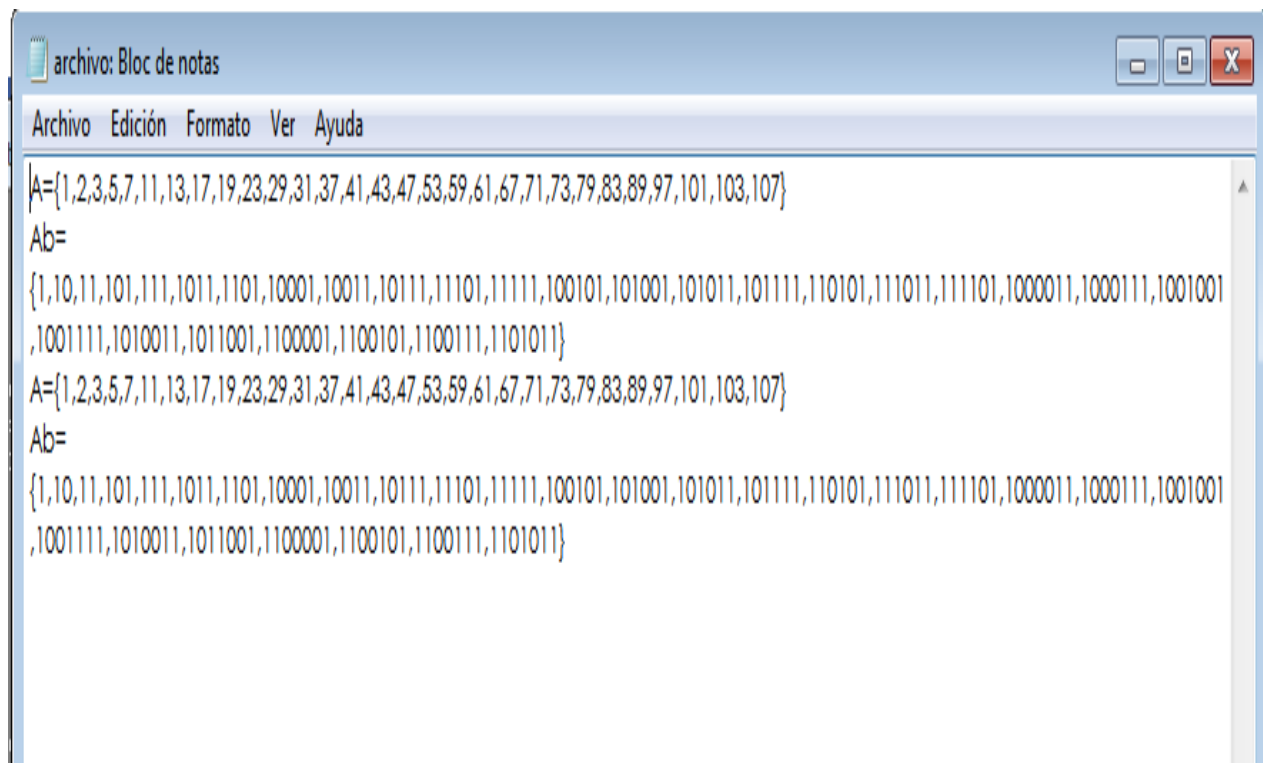


Figura 5: Archivo resultante en ambas opciones

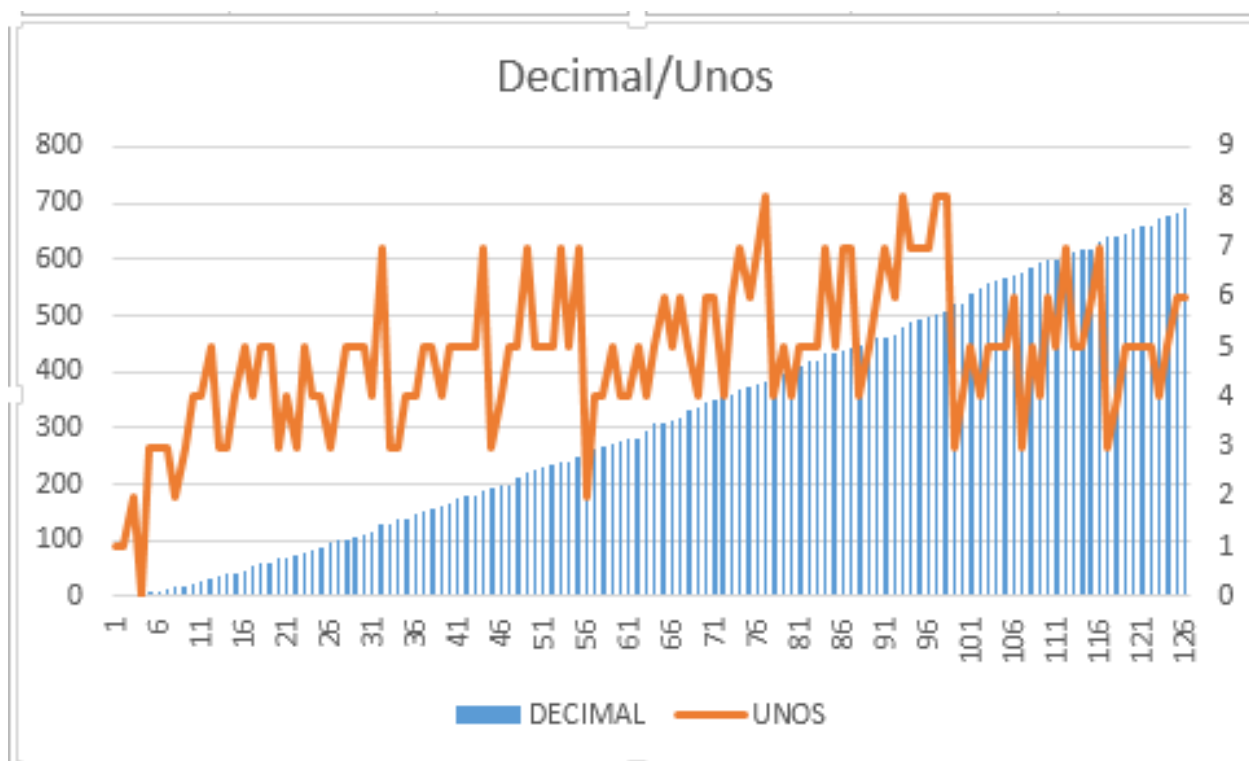
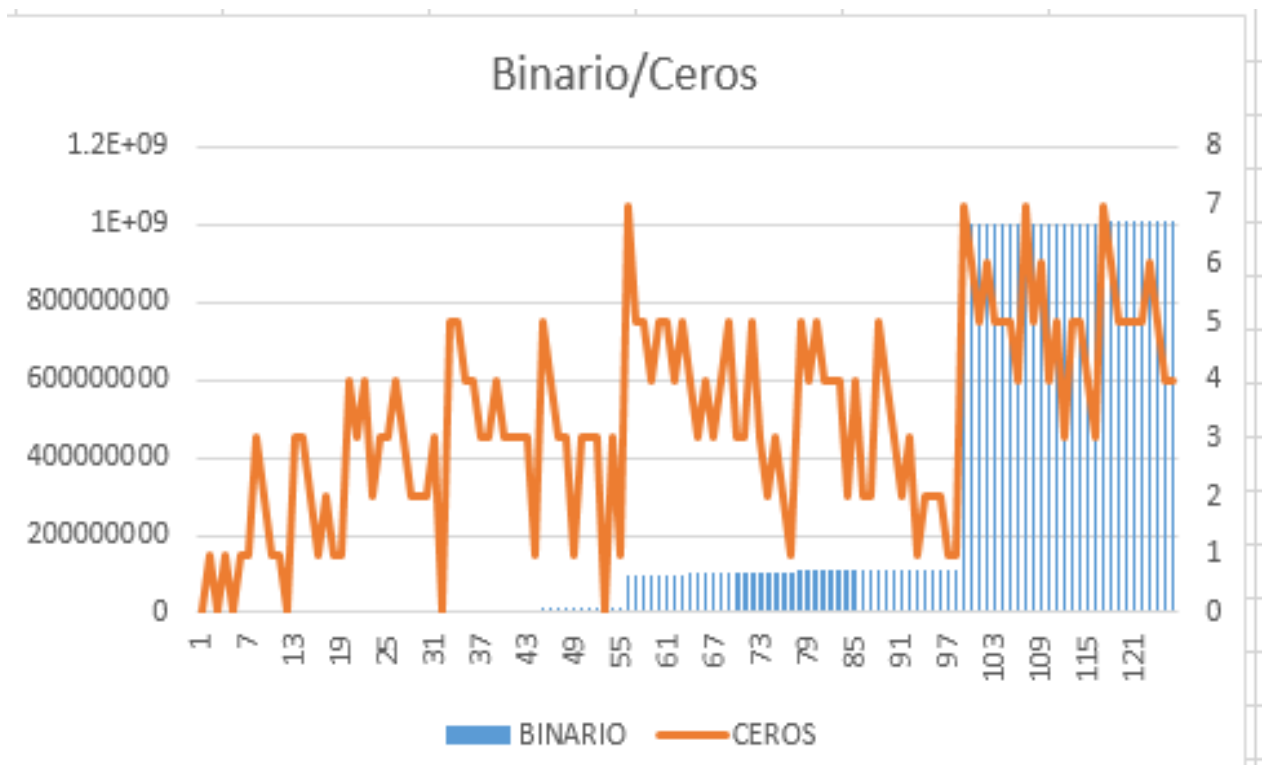


Figura 6: Gráfica



4. Autómata terminación -ere

Los autómatas evalúan a través de estados, un autómata determinístico es aquel que según la entrada asigna un valor, esto quiere decir que todas sus opciones son fijas, este es un autómata sencillo y que ayuda a la comprensión de que son los autómatas y comenzar a implementarlos de una forma sencilla.

4.1. Descripción

Crear un autómata que reconoce las cadenas con terminación *ere*. Después vamos almacenar las cadenas validadas, su ubicación y también la ruta que tomo al evaluar las cadenas, debe de ser capaz de analizar todas las posibles opciones que se presenten.

4.2. Código

Código fuente del autómata:

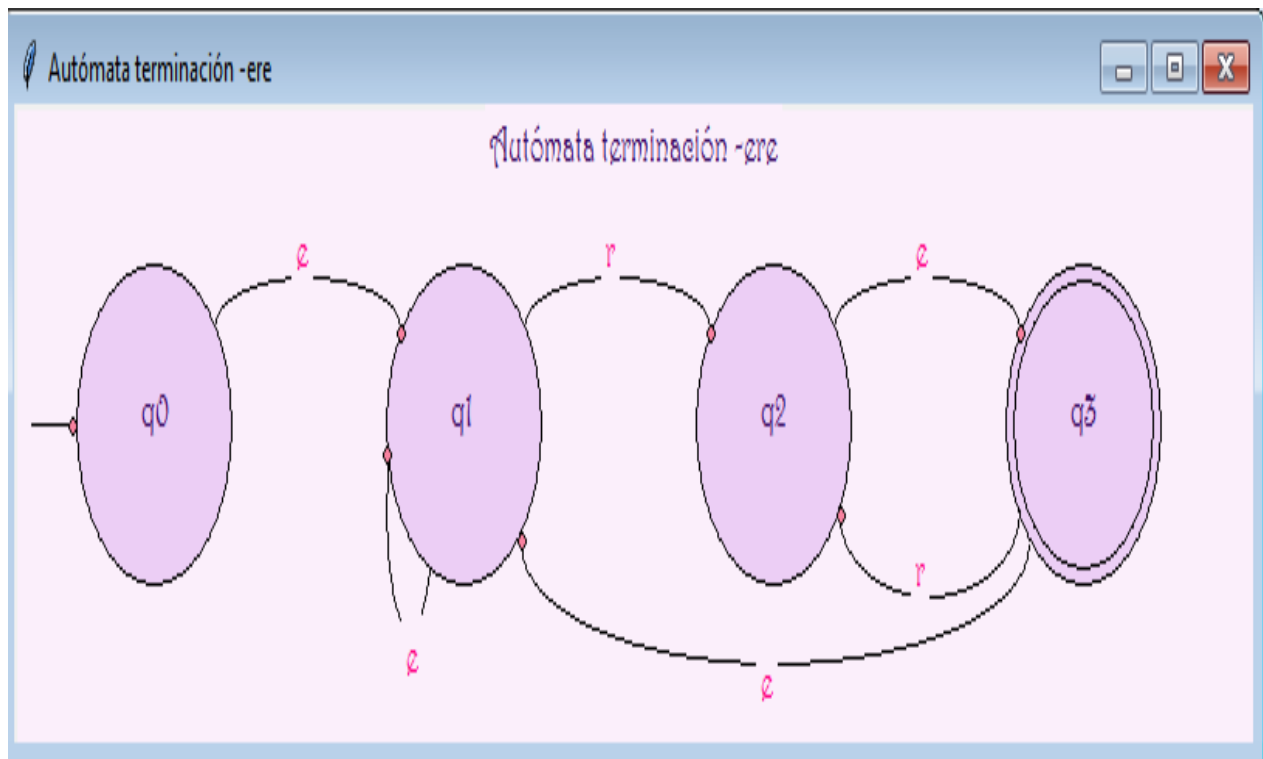


Figura 7: El modelodel autómata es:

El lenguaje utilizado para este programa fue Python

Archivo: ere.py

```
import random
from tkinter import *
def grafico ():
    ventana=Tk()
    g=Canvas(ventana,bg="#FBEFFB",width=800, height=200)
    g.pack()
    ventana.title("Aut mata_terminaci n_ere_")
    ventana.geometry("800x200")
    color="#ECCEF5"
    color1="#380B61"
    etiqueta = Label(ventana, bg="#FBEFFB",text="Aut mata_terminaci n_ere",fg=color1, font="Harrington")
    g.create_oval(40,50,140,150, fill=color)
    etiqueta1=Label(ventana, bg=color,text="q0",fg=color1, font="Harrington").place(x=79,y=83)
    g.create_oval(240,50,340,150, fill=color)
    etiqueta2=Label(ventana, bg=color,text="q1",fg=color1, font="Harrington").place(x=279,y=83)
    g.create_oval(440,50,540,150, fill=color)
    etiqueta3=Label(ventana, bg=color,text="q2",fg=color1, font="Harrington").place(x=479,y=83)
    g.create_oval(640,50,740,150, fill=color)
    g.create_oval(645,55,735,145, fill=color)
    etiqueta4=Label(ventana, bg=color,text="q3",fg=color1, font="Harrington").place(x=679,y=83)
    color3="#FF0080"
    color4="#F7819F"
    g.create_arc(10,100,40,100, start=0, extent =180, style='arc')
    g.create_oval(35,97.5,40,102.5, fill=color4)
    g.create_arc(129,85,249,54, start=360, extent =180, style='arc')
    g.create_arc(329,85,449,54, start=360, extent =180, style='arc')
    g.create_arc(529,85,649,54, start=360, extent =180, style='arc')
    g.create_arc(532,100,649,154, start=360, extent =-180, style='arc')
    g.create_arc(327,100,655,175, start=360, extent =-180, style='arc')
    g.create_arc(240,90,270,164, start=332, extent =-180, style='arc')
    g.create_oval(246.5,68.5,251.5,73.5, fill=color4)
    g.create_oval(446.5,68.5,451.5,73.5, fill=color4)
    g.create_oval(646.5,68.5,651.5,73.5, fill=color4)
    g.create_oval(237.5,106.5,242.5,111.5, fill=color4)
    g.create_oval(324.5,133.5,329.5,138.5, fill=color4)
    g.create_oval(530.5,125.5,535.5,130.5, fill=color4)
    etiqueta4=Label(ventana, bg="#FBEFFB",text="e",fg=color3, font="Harrington").place(x=579,y=33)
    etiqueta4=Label(ventana, bg="#FBEFFB",text="r",fg=color3, font="Harrington").place(x=379,y=33)
    etiqueta4=Label(ventana, bg="#FBEFFB",text="e",fg=color3, font="Harrington").place(x=179,y=33)
    etiqueta4=Label(ventana, bg="#FBEFFB",text="r",fg=color3, font="Harrington").place(x=579,y=133)
```

```

etiqueta4=Label(ventana, bg="#FBEFFB",text="e",fg=color3, font="
    Harrington").place(x=479,y=168)
etiqueta4=Label(ventana, bg="#FBEFFB",text="e",fg=color3, font="
    Harrington").place(x=250,y=160)
g.place(x=0,y=0)
etiqueta.pack()
g.mainloop()
def manual():
    cadena=""
    ubi=open("Camino.txt","w")
    ubi.close()
    Coordinadas=open("Coordinadas.txt","w")
    Coordinadas.close()
    print("\t_Usted_eligio_el_modo_manual\n_Ingrese_la_cadena_
        porfavor")
    cadena=input()
    archivo=open("Entrada.txt","w")
    archivo.write(cadena)
    archivo.close()
    fp=open("Validado.txt","w")
    fp.close()
    revision(1)
def automatico():
    cadena=""
    ubi=open("Camino.txt","w")
    ubi.close()
    Coordinadas=open("Coordinadas.txt","w")
    Coordinadas.close()
    print("\t_Usted_eligio_el_modo_automatico")
    archivo=open("Automatico.txt","r")
    archivo.read()
    archivo.close()
    fp=open("Validado.txt","w")
    fp.close()
    revision(2)
def revision(j):
    if j==1:
        archivo=open("Entrada.txt","r")
    elif j==2:
        archivo=open("Automatico.txt","r")
    y=0
    for linea in archivo:
        cadena=linea
        estados(cadena,y)
        y=y+1
    archivo.close()
def estados(cadena,y):
    ubi=open("Camino.txt","a")
    Coordinadas=open("Coordinadas.txt","a")
    caracter=0
    palabra=cadena.split()

```

```

x=0
banl=0
for i in palabra:
    banl=0
    tama o=len(i)
    estado=0
    c=0
    for a in i:
        if estado==0:
            ubi.write("q0_" + a + " -->")
            if a=="e" or a=="E":
                estado=1
            else:
                estado=0
        elif estado==1:
            ubi.write("q1_" + a + " -->")
            if a=="r" or a=="R":
                estado=2
            elif a=="e" or a=="E":
                estado=1
            else :
                estado=0
        elif estado==2:
            ubi.write("q2_" + a + " -->")
            if a=="e" or a=="E":
                estado=3
            else:
                estado=0
        elif estado==3:
            if tama o==c:
                ubi.write("q3_" + a + "FIN")
                banl=1
                guardar(i)
                Coordenadas.write("Ubicacion:" + str(y
                    +1) + "," + str(x+1) + "," + str(
                    caracter-c) + "\n")
            elif tama o==c+1:
                estado=5
            elif a=="r" or a=="R":
                ubi.write("q3_" + a + " -->")
                estado=2
            elif a=="e" or a=="E":
                ubi.write("q3_" + a + " -->")
                estado=1
            else:
                estado=0
        elif estado==5:
            if a==" ," or a==" ." or a=="?" or a=="!" or a=="
                "}" or a==" ]" or a==" )" or a==" '" :
                guardar(i[0:len(i)-1])
                ubi.write("q3_" + a + "FIN")

```

```

        banl=1
        Coordenadas.write("Ubicacion:" + str(y
            +1) + "," + str(x+1) + "," + str(
                caracter-c) + "\n")

        c=c+1
        if estado==3:
            if tamano==c:
                ubi.write("q3_-" + a + "FIN")
                banl=1
                guardar(i)
                Coordenadas.write("Ubicacion:" + str(y
                    +1) + "," + str(x+1) + "," + str(
                        caracter-c) + "\n")
            elif tamano==c+1:
                estado=5
            else:
                estado=0
        elif estado==5:
            if a==" ," or a==" ." or a=="?" or a=="!" or a=="
                "}" or a=="]" or a==" )" or a==" '" :
                guardar(i[0:len(i)-1])
                ubi.write("q3_-" + a + "FIN")
                banl=1
                Coordenadas.write("Ubicacion:" + str(y
                    +1) + "," + str(x+1) + "," + str(
                        caracter-c) + "\n")

        if banl==0:
            ubi.write("No_Valida\n")
        else:
            ubi.write("\n")
        x=x+1
        caracter=caracter+c
def guardar(i):
    palabra=i
    valida=open ("Validado.txt","a")
    valida.write(palabra)
    valida.write("_")
    valida.close()
def menu():
    entra=1
    while entra==1:
        print("\t\tSelecione_la_opcion_deseada\n1.-Manual\n2.-
            Automatico\n3.- Grafico\n")
        #opc=str(input())
        opc=str(random.randrange(1,4))
        print(opc)
        if opc=='1':
            manual()
        elif opc=='2':
            automatico()
        elif opc=='3':

```

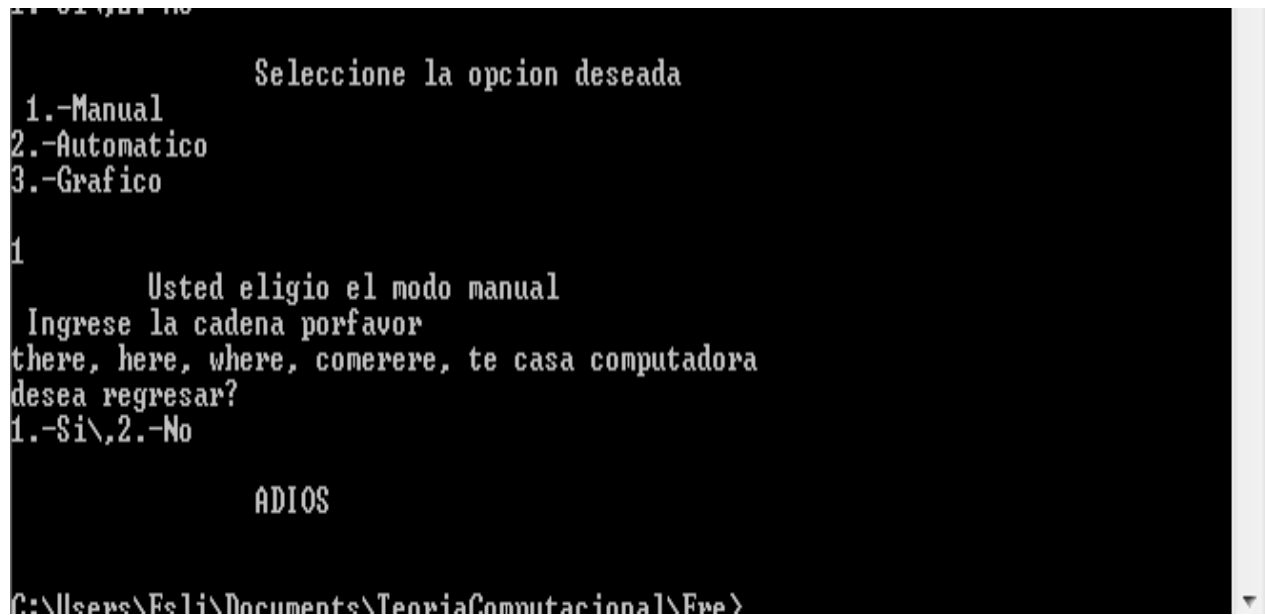
```

        print("\t\tFue_seleccionado_el_modo_Grafico\n")
        grafico()
    print("desea_regresar?\n1.-Si\,2.-No\n")
    #entra=int(input())
    entra=random.randrange(0,2)
    print("\t\tADIOS\n")
menu()

```

4.3. Pruebas

Resultados.



```

Seleccione la opcion deseada
1.-Manual
2.-Automatico
3.-Grafico
1
    Usted eligio el modo manual
    Ingrese la cadena porfavor
there, here, where, comerere, te casa computadora
desea regresar?
1.-Si\,2.-No
    ADIOS
C:\Users\Eslu\Documents\TeoriaComputacional\Fre>

```

Figura 8: consola manual-automático.

Figura 9: Un texto automático.

Figura 10: Un archivo generado de modo manual.

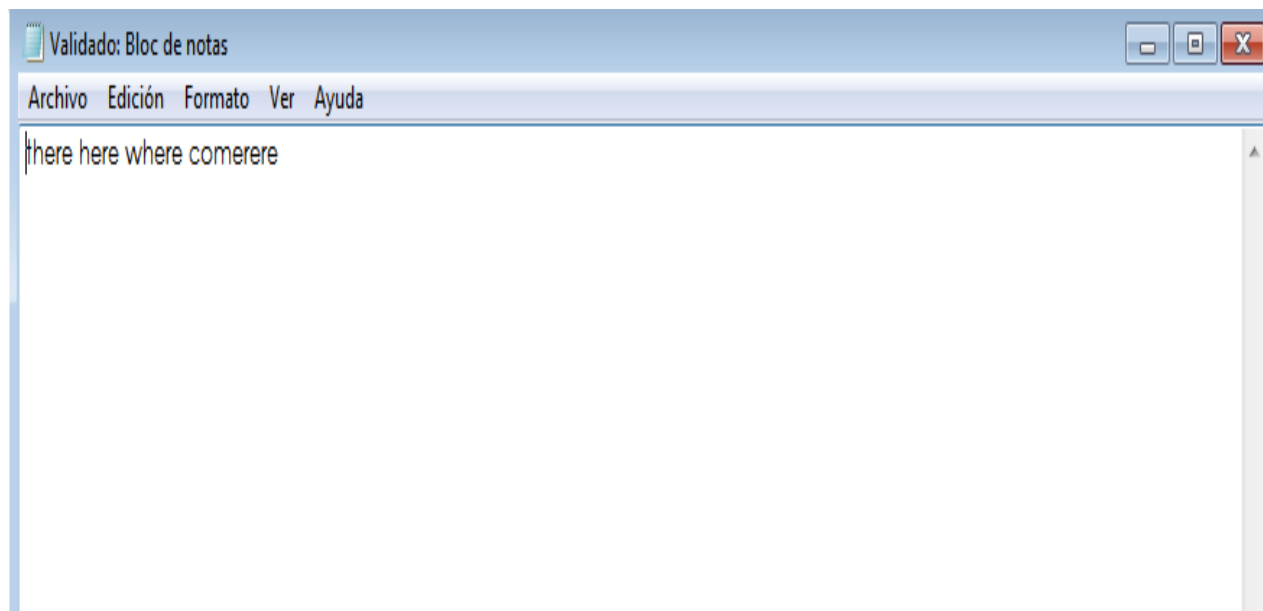


Figura 11: Archivo de palabras validadas.

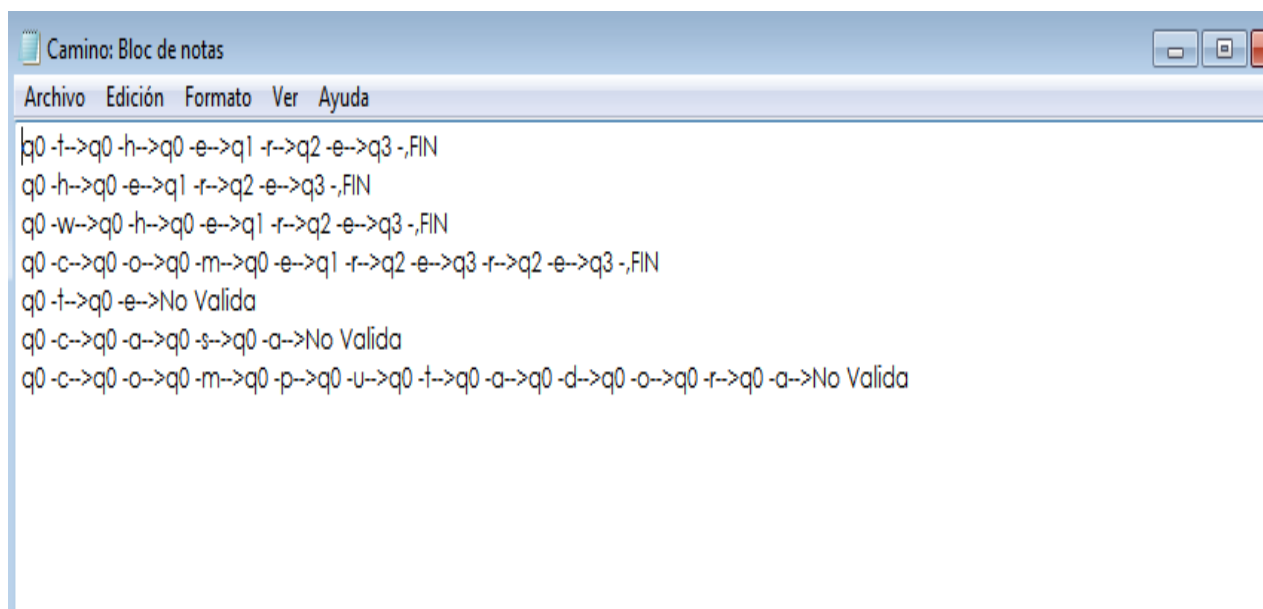


Figura 12: Archivo de ruta.

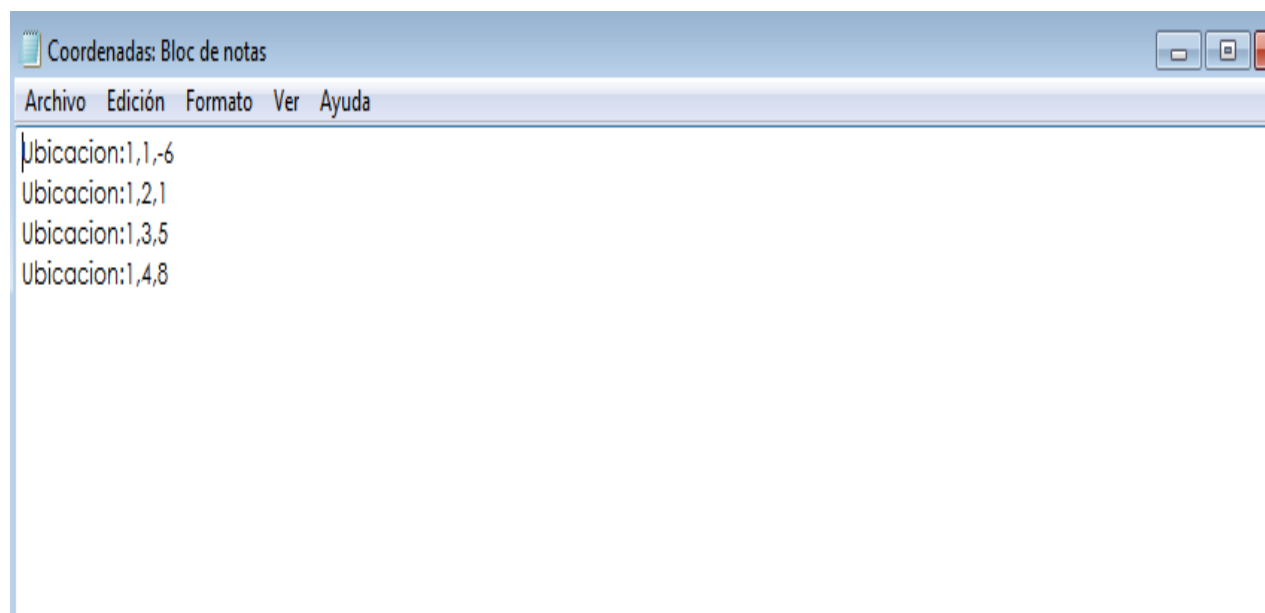


Figura 13: Archivo de coordenadas.

5. Autómata de paridad

Este autómata sigue siendo determinístico, al ingresar una cadena a de 1's y 0's realmente no sabemos que cantidad de 1's y 0's contiene, no tenemos tampoco una fórmula para contarlos, solamente a mano, para solucionar esto implementaremos un autómata.

5.1. Descripción

Crear un autómata que nos verifique si una cadena tiene paridad o sea que su número de 0s y 1's sea par. Después las cadenas se guardan en un archivo colocándonos ahí cuales son las que tiene paridad y en otro archivo se guardan los estados.

5.2. Código

Código fuente:

Figura 14: El autómata utilizado para este problema

El lenguaje utilizado para este programa fue Python

Archivo: paridad.py

```
import random
from tkinter import *
def grafico():
    ventana=Tk()
    g=Canvas(ventana,bg="#FBEFFB",width=500, height=500)
    g.pack()
    ventana.title("Aut mata_de_Paridad_")
    ventana.geometry("400x400")
    color="#ECCEF5"
    color1="#380B61"
    etiqueta = Label(ventana, bg="#FBEFFB",text="Aut mata_de_Paridad",fg=
        color1, font="Harrington")
    g.create_oval(40,50,140,150, fill=color)
    g.create_oval(45,55,135,145, fill=color)
    etiqueta1=Label(ventana, bg=color, text="q0",fg=color1, font="
        Harrington").place(x=79,y=83)
    g.create_oval(240,50,340,150, fill=color)
    etiqueta2=Label(ventana, bg=color, text="q1",fg=color1, font="
        Harrington").place(x=279,y=83)
    g.create_oval(240,250,340,350, fill=color)
    etiqueta3=Label(ventana, bg=color, text="q2",fg=color1, font="
        Harrington").place(x=79,y=283)
```

```

g.create_oval(40,250,140,350, fill=color)
etiqueta4=Label(ventana, bg=color, text="q3", fg=color1, font="
    Harrington").place(x=279,y=283)
color3="#FF0080"
color4="#F7819F"
g.create_arc(10,100,40,100, start=0, extent =180, style='arc')
g.create_oval(35,97.5,40,102.5, fill=color4)
g.create_arc(129,85,249,54, start=360, extent =180, style='arc')#
    arriba
g.create_arc(129,115,249,146, start=360, extent =-180, style='arc')#
    abajo
g.create_arc(240,132,270,265, start=90, extent =180, style='arc')#
    izquierda
g.create_arc(310,132,340,265, start=90, extent =-180, style='arc')#
    derecha
g.create_arc(129,286,249,255, start=360, extent =180, style='arc')#
    arriba2
g.create_arc(129,316,251,347, start=360, extent =-180, style='arc')#
    abajo2
g.create_arc(45,132,65,265, start=90, extent =180, style='arc')#
    izquierda
g.create_arc(115,132,135,265, start=90, extent =-180, style='arc')#
    derecha
g.create_oval(246.5,68.5,251.5,73.5, fill=color4)
g.create_oval(127,128.5,132,133.5, fill=color4)
g.create_oval(325.5,262.5,330.5,267.5, fill=color4)
g.create_oval(251.5,131.5,256.5,136.5, fill=color4)
g.create_oval(246.5,268.5,251.5,273.5, fill=color4)
g.create_oval(127,329.5,132,334.5, fill=color4)
g.create_oval(51.5,261.5,57.5,266.5, fill=color4)
g.create_oval(123.5,131.5,128.5,136.5, fill=color4)
etiqueta4=Label(ventana, bg="#FBEFFB", text="1", fg=color3, font="
    Harrington").place(x=179,y=38)
etiqueta4=Label(ventana, bg="#FBEFFB", text="1", fg=color3, font="
    Harrington").place(x=179,y=133)
etiqueta4=Label(ventana, bg="#FBEFFB", text="0", fg=color3, font="
    Harrington").place(x=232,y=190)
etiqueta4=Label(ventana, bg="#FBEFFB", text="0", fg=color3, font="
    Harrington").place(x=332,y=193)
etiqueta4=Label(ventana, bg="#FBEFFB", text="0", fg=color3, font="
    Harrington").place(x=179,y=238)
etiqueta4=Label(ventana, bg="#FBEFFB", text="0", fg=color3, font="
    Harrington").place(x=179,y=333)
etiqueta4=Label(ventana, bg="#FBEFFB", text="1", fg=color3, font="
    Harrington").place(x=40,y=185)
etiqueta4=Label(ventana, bg="#FBEFFB", text="1", fg=color3, font="
    Harrington").place(x=130,y=185)
g.place(x=0,y=0)
etiqueta.pack()
g.mainloop()

```

def manual():

```

cadena=""
ubi=open ("Camino.txt","w")
ubi.close()
print ("\t_Usted_eligio_el_modo_manual\n_Ingrese_la_cadena_de_
      i's_y_0's_porfavor")
cadena=input ()
archivo=open ("Manual.txt","w")
archivo.write(cadena)
archivo.close()
fp=open ("Validado.txt","w")
fp.close()
revision(1)
def automatico():
    cadena=""
    ubi=open ("Camino.txt","w")
    ubi.close()
    print ("\t_Usted_eligio_el_modo_automatico")
    archivo=open ("Automatico.txt","r")
    archivo.read()
    archivo.close()
    fp=open ("Validado.txt","w")
    fp.close()
    revision(2)
def revision(j):
    if j==1:
        archivo= open("Manual.txt","r")
    elif j==2:
        archivo= open("Automatico.txt","r")
    for linea in archivo:
        cadena=linea
        estados(cadena)
    archivo.close()
def estados(cadena):
    ubi=open ("Camino.txt","a")
    palabra=cadena.split()
    ban2=0
    ban1=0
    for i in palabra:
        ban1=0
        ban2=0
        tamaño=len(i)
        estado=0
        c=0
        for a in (i+","):
            if estado==0:
                ubi.write("q0_-"+a+"-->")
                if a=="0":
                    estado=2
                elif a=="1":
                    estado=1
                elif tamaño==c:

```

```

        if a=="," or a=="." :
            if ban2==0:
                ban1=1
                ubi.write("FIN\n")
                guardar(i[0:len(i)])
            else :
                ban1=1
                ubi.write("FIN\n")
                guardar(i[0:len(i)-1])
    elif tama o==c+1:
        if a=="," or a=="." :
            estado=0
            ban2=1
    elif estado==1:
        ubi.write("q1_-"+a+"-->")
        if a=="0":
            estado=3
        elif a=="1":
            estado=0
    elif estado==2:
        ubi.write("q2_-"+a+"-->")
        if a=="0":
            estado=0
        elif a=="1":
            estado=3
    elif estado==3:
        if a=="0":
            estado=1
        elif a=="1":
            estado=2
    if ban1==0 and tama o==c:
        ubi.write("No_Valida\n")
        c=c+1

def guardar(i):
    palabra=i
    valida=open("Validado.txt","a")
    valida.write(palabra)
    valida.write("\n")
    valida.close()

def menu():
    entra=1
    while entra==1:
        print("\t\t\tSeleccione_la_opcion_deseada\n1.-Manual\n2.-
Automatico\n3.- Grafico\n")
        opc=str(input())
        opc=str(random.randrange(1,4))
        if opc=='1':
            manual()
        elif opc=='2':
            automatico()
        elif opc=='3':

```

```

        print("\t\tFue_seleccionado_el_modulo_Grafico\n")
        grafico()
    print("desea_regresar?\n1.- Si\,2.- No\n")
    #entra=int(input())
    entra=random.randrange(0,2)
    print("\t\tADIOS\n")
menu()

```

5.3. Pruebas

Resultados.

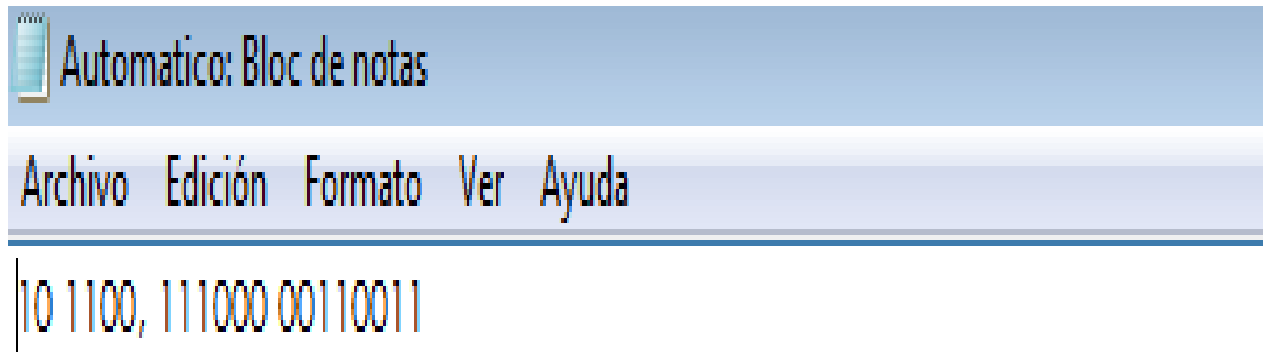


Figura 15: Un texto automático.

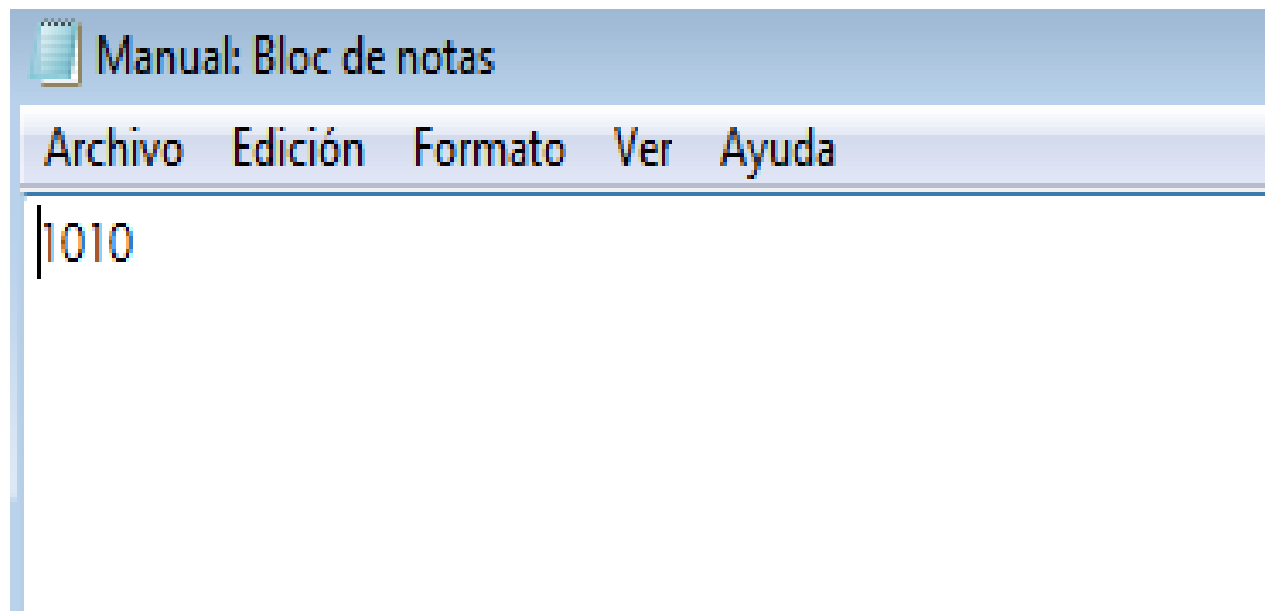


Figura 16: Un archivo generado de modo manual.

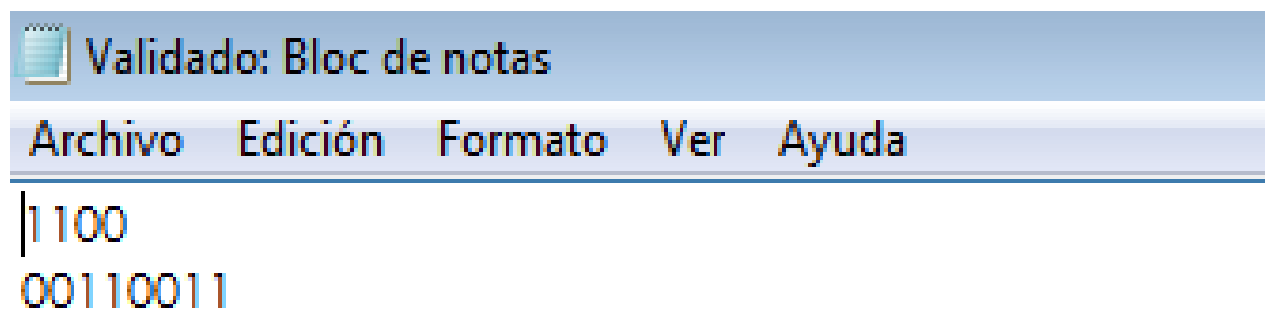
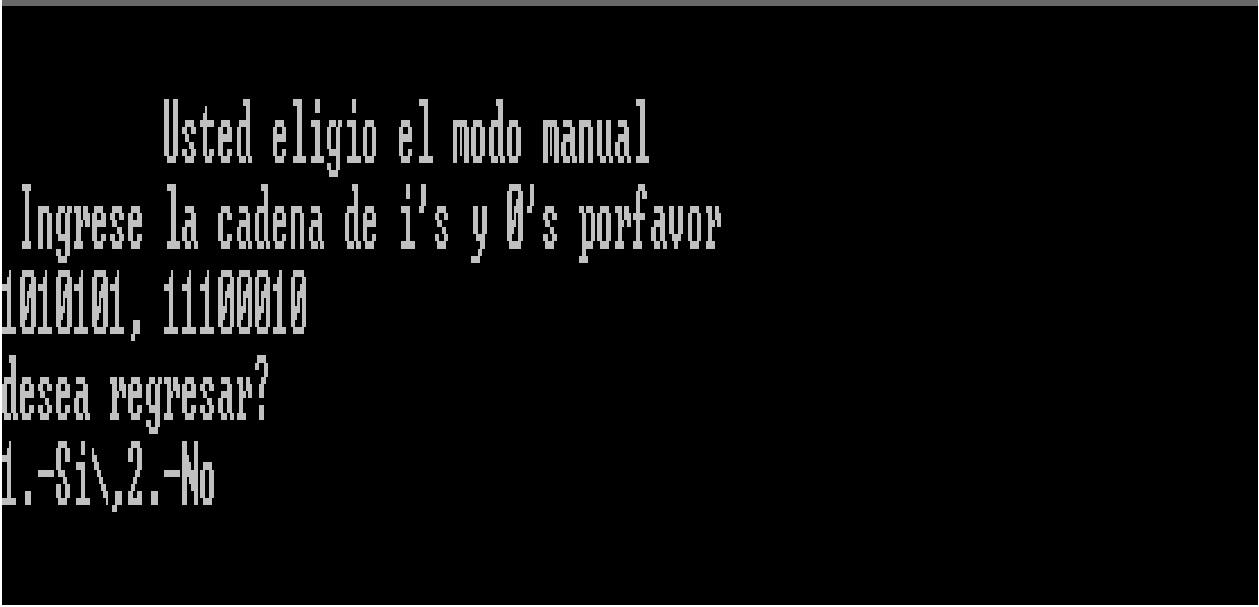


Figura 17: Archivo de palabras validadas.



```
Usted eligio el modo manual
Ingrese la cadena de i's y 0's porfavor
1010101, 11100010
desea regresar?
1.-Si\,2.-No
```

Figura 18: Consola manual-automático.

6. Protocolo

Este es un protocolo, es un programa totalmente automático, que evalúa tramas de datos.

6.1. Descripción

Crear un programa protocolo totalmente automático que en primer lugar pregunte si esta encendido el receptor, si esta encendido crea 50 tramas de 0's y 1's, ya que los recibe el receptor se espera un segundo y pasa por al autómata de paridad y regresa las cadenas validadas y así comienza de nuevo el ciclo hasta que está apagado el receptor.

6.2. Código

Código fuente:

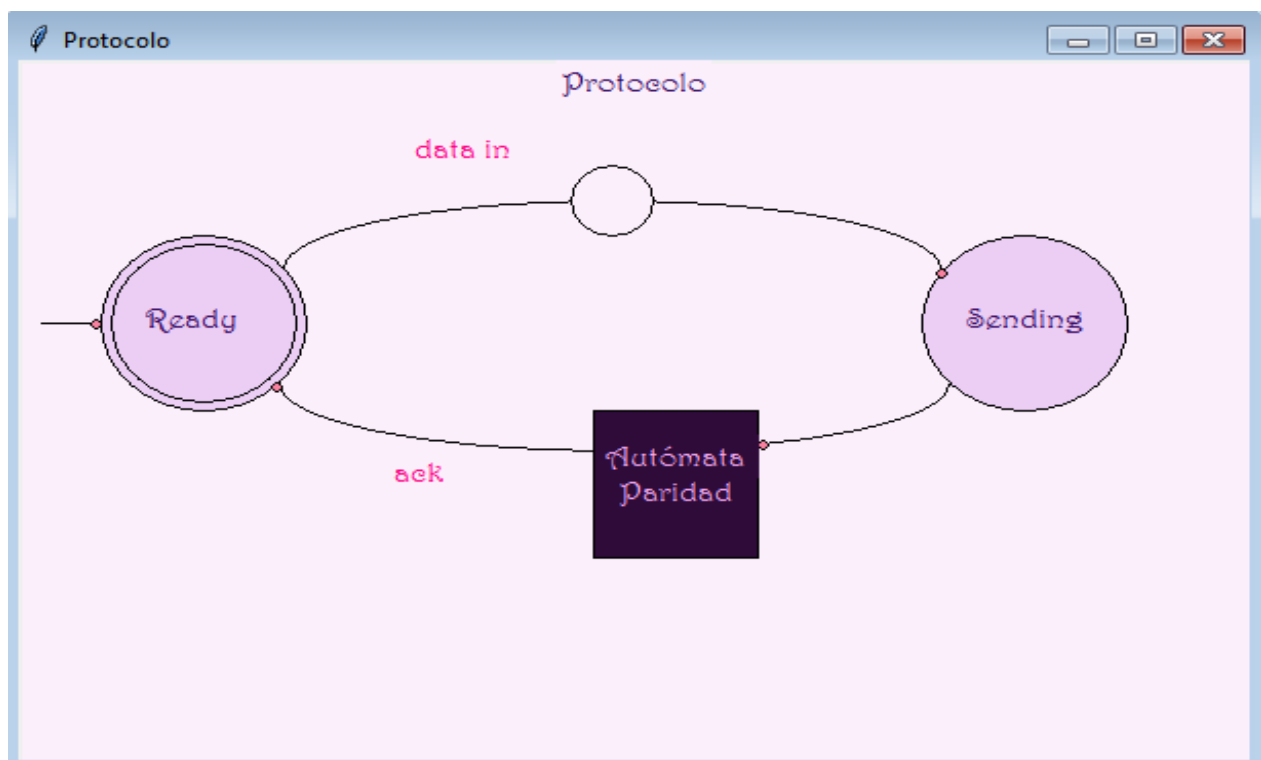


Figura 19: El modelo utilizado para este problema

El lenguaje utilizado para este programa fue Python

Archivo: protocolo.py

```

import random
import time
from tkinter import *
def grafico(encendido):
    ventana=Tk()
    g=Canvas(ventana,bg="#FBEFFB",width=600, height=600)
    g.pack()
    ventana.title("Protocolo")
    ventana.geometry("600x400")
    color="#ECCEF5"
    color1="#380B61"
    etiqueta = Label(ventana, bg="#FBEFFB",text="Protocolo",fg=color1,
        font="Harrington")
    g.create_oval(40,100,140,200, fill=color)
    g.create_oval(45,105,135,195, fill=color)
    etiqueta1=Label(ventana, bg=color, text="Ready",fg=color1, font="
        Harrington").place(x=59,y=135)
    g.create_oval(440,100,540,200, fill=color)
    etiqueta2=Label(ventana, bg=color, text="Sending",fg=color1, font="
        Harrington").place(x=459,y=135)
    color3="#FF0080"
    color4="#F7819F"
    g.create_arc(10,150,40,150, start=0, extent =180, style='arc')
    g.create_oval(35,147.5,40,152.5, fill=color4)
    g.create_arc(129,158,449,80, start=360, extent =180, style='arc')#
        arriba
    g.create_arc(127,146,453,223, start=360, extent =-180, style='arc')#
        abajo2
    g.create_oval(446.5,118.5,451.5,123.5, fill=color4)
    g.create_oval(122.5,188.5,127.5,183.5, fill=color4)
    g.create_oval(360,216.5,365,221.5, fill=color4)
    etiqueta4=Label(ventana, bg="#FBEFFB",text="data_in",fg=color3, font="
        Harrington").place(x=190,y=38)
    etiqueta4=Label(ventana, bg="#FBEFFB",text="ack",fg=color3, font="
        Harrington").place(x=180,y=223)
    if encendido==1:
        g.create_oval(269,60,309,100, fill="#FF0040")
    elif encendido==0:
        g.create_oval(269,60,309,100, fill="#FBEFFB")
    g.create_rectangle(280,200,360,284, fill="#2F0B3A")
    etiqueta4=Label(ventana, bg="#2F0B3A",text="Aut mata_",fg="#F5A9F2",
        font="Harrington").place(x=283,y=214)
    etiqueta4=Label(ventana, bg="#2F0B3A",text="Paridad_",fg="#F5A9F2",
        font="Harrington").place(x=290,y=234)
    g.place(x=0,y=0)
    etiqueta.pack()
    g.mainloop()
def creacion():
    archivo=open("Automatico.txt","w")
    for a in range(50):

```

```

        cadena=""
        for i in range(32):
            cadena+=str(random.randrange(0,2))
        archivo.write(str(a)+".-"+cadena+"\n")
def revision():
    archivo= open("Automatico.txt","r")
    for linea in archivo:
        cadena=linea
        estados(cadena)
    archivo.close()
def estados(cadena):
    ubi=open("Camino.txt","a")
    palabra=cadena.split()
    ban2=0
    ban1=0
    for i in palabra:
        ban1=0
        ban2=0
        tamaño=len(i)
        estado=0
        c=0
        for a in (i+","):
            if estado==0:
                ubi.write("q0_"+a+"-->")
                if a=="0":
                    estado=2
                elif a=="1":
                    estado=1
                elif tamaño==c:
                    ban1=1
                    ubi.write("FIN\n")
                    guardar(i)
            elif estado==1:
                ubi.write("q1_"+a+"-->")
                if a=="0":
                    estado=3
                elif a=="1":
                    estado=0
            elif estado==2:
                ubi.write("q2_"+a+"-->")
                if a=="0":
                    estado=0
                elif a=="1":
                    estado=3
            elif estado==3:
                if a=="0":
                    estado=1
                elif a=="1":
                    estado=2
        if ban1==0 and tamaño==c:
            ubi.write("No_Valida\n")

```

```

c=c+1
def guardar(i):
    palabra=i
    valida=open ("Validado.txt","a")
    valida.write(palabra)
    valida.write("\n")
    valida.close()
def menu():
    archivo=open ("Automatico.txt","w")
    encendido=1
    while encendido==1:
        encendido=random.randrange(0,2)
        camprot=open ("camprot.txt","a")
        if encendido==1:
            grafico(1)
            print ("\t\tEncendido\n")
            camprot.write("Encendido_>_")
            creacion()
            camprot.write("Creacion_>_")
            ubi=open ("Camino.txt","w")
            ubi.close()
            fp=open ("Validado.txt","w")
            fp.close()
            archivo=open ("Automatico.txt","r")
            archivo.read()
            archivo.close()
            print ("\t\tRevision_en_curso\n")
            camprot.write("Revision_>_Fin\n")
            time.sleep(2)
            revision()
        else:
            grafico(0)
            camprot.write("No_encendido_\n")
            print ("\t\tNo_esta_encendido\n")
            encendido=random.randrange(0,2)
        print ("\t\tYa_no_esta_encendido\n")
        camprot.write("Ya_no_esta_encendido_\n")

menu()

```

6.3. Pruebas

Imagenes del Programa ejecutandose.

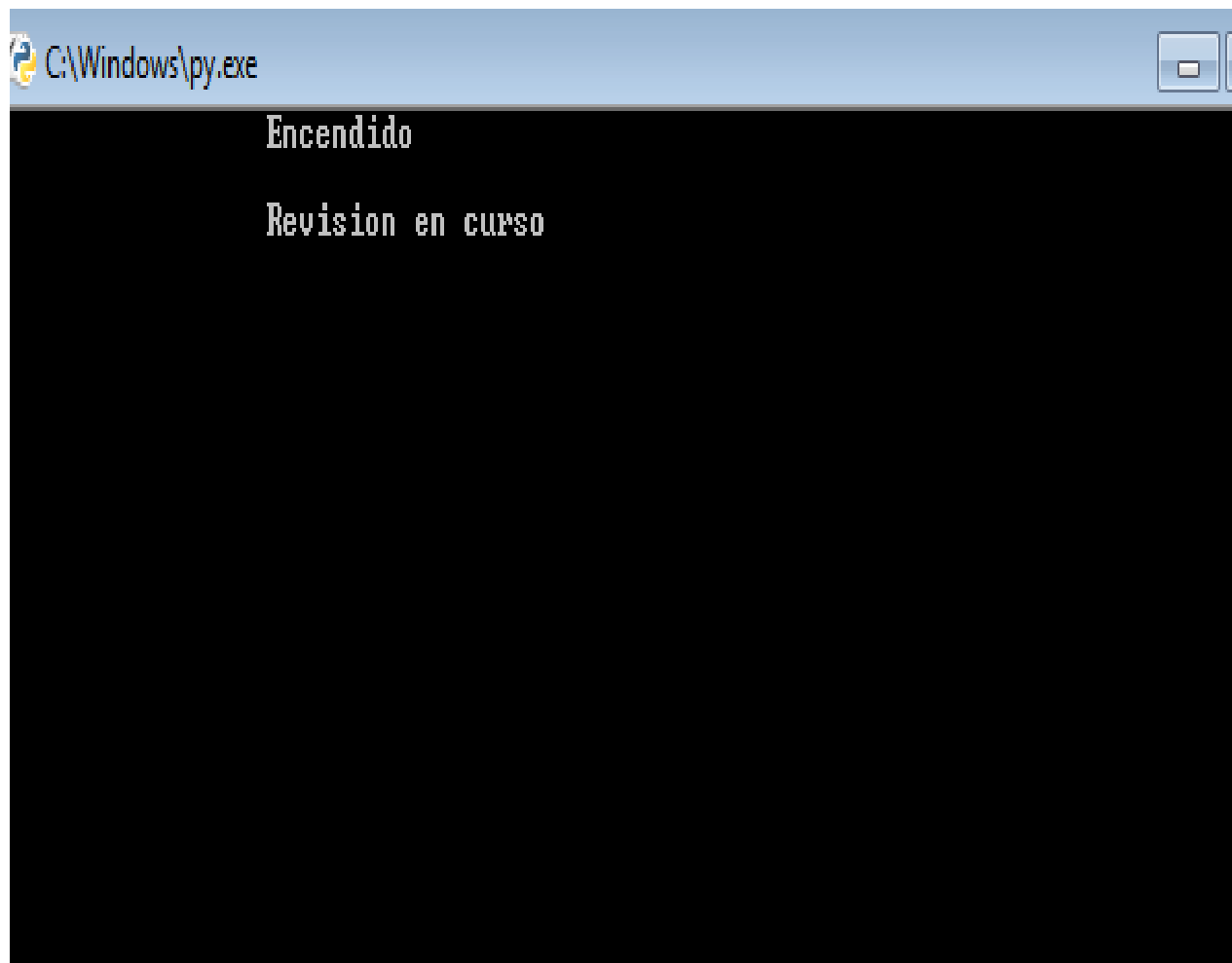


Figura 20: consola

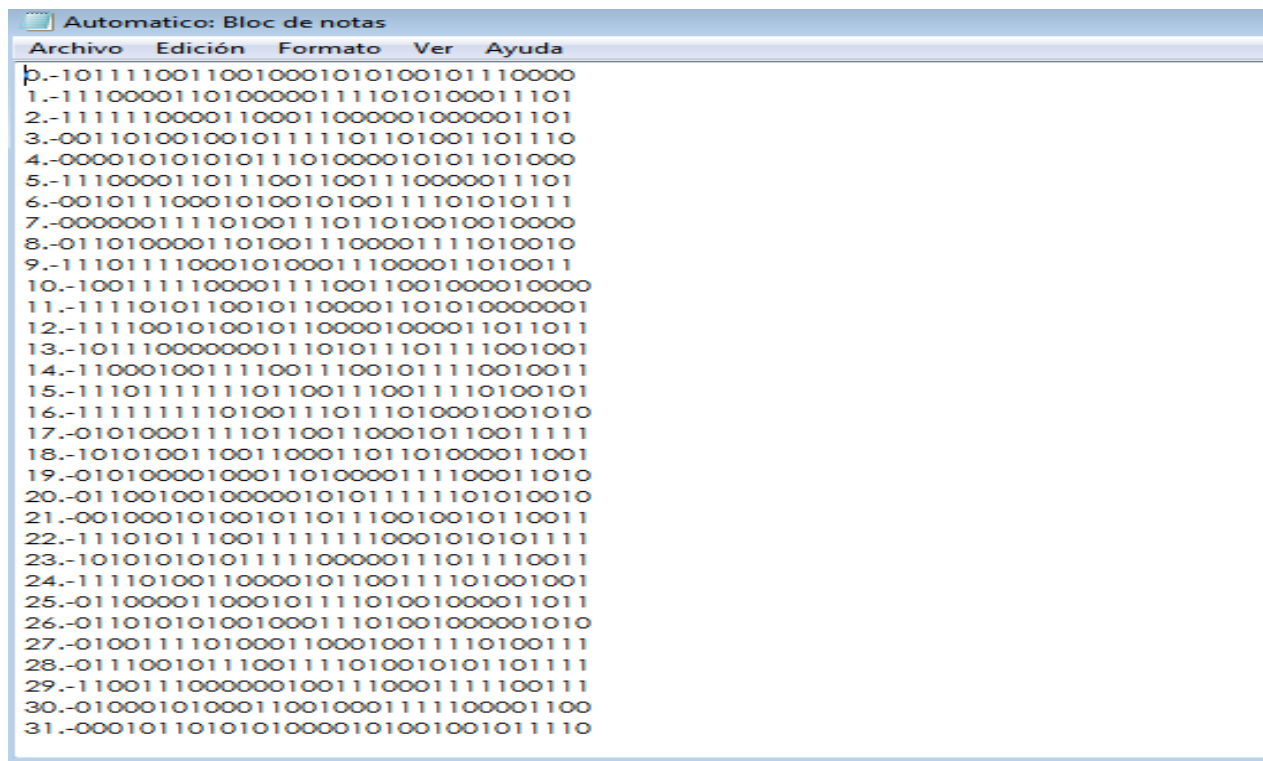


Figura 21: Archivo generado automaticamente

Figura 22: Camino con las evaluaciones

Validado: Bloc de notas

Archivo Edición Formato Ver Ayuda

2.-11111100001100011000001000001101
3.-00110100100101111101101001101110
22.-11101011100111111110001010101111
28.-01110010111001111010010101101111
34.-10011011101000010001110000110111
36.-10000001011101001111101011110110
38.-01111000000010101000010111110111
39.-01110001111000111111101110111101
42.-11101101011110100011110101111101
47.-11101000010001100000010000100010

7. Cadenas que terminan en 01

Este programa no supe como hacerlo, investigue y lo intente pero realmente no supe muy bien como realizarlo

Referencias

"Teoría de Autoómatas y lenguajes de programación", Hopcrof J., Motwain R., Vlimn J., Addison Wesley , 2008