

REPORTE

Programa 7 - palindromes

Materia : Teoría de la Computación

Grupo : 4CM1

Alumno : Julio Cesar Hernández Reyes

Docente : Juárez Martínez Genaro

1. Introducción

En este reporte se explica como se realizo el programa 7 - palindromes , el cual es un programa que tiene la opción de crear de forma manual o automatica palindromes de un lenguaje binario usando las 5 reglas de producción siguientes:

- (1) $P \rightarrow e$
- (2) $P \rightarrow 0$
- (3) $P \rightarrow 1$
- (4) $P \rightarrow 0P0$
- (5) $P \rightarrow 1P1$

La entrada de la opción manual es la longitud que se requiera que el palindromo tenga, despues se hace el proceso de creacion de esa cadena usando aleatoriamente las reglas de produccion antes mencionadas. La salida de este programa es el archivo Historia de como se fue creando el palindromo asi como el palindromo final.

En la forma automática el valor máximo que puede ser un palindromo es de 100,000 caracteres.

Para la realización del programa se uso Python. Se uso un IDE en vez de compilar y ejecutar por consola, esto para una mayor facilidad a la hora de corregir errores y algunos detalles del programa.



PyCharm Community Edition
2021.2.1
Aplicación

2. Información Importante

El funcionamiento para la creación de los palindromes es algo sencillo, una vez teniendo la longitud requerida para la cadena a crear, se tienen dos opciones posibles, que la cadena sea par o impar.

En base en lo anterior se tiene que si es par solo se tiene que iterar las veces igual a la mitad de la longitud, haciendo uso de las reglas 4 y 5 aleatoriamente, para al final usar la regla 1 y que el palindromo tenga la longitud requerida.

La otra parte es cuando se requiera que la cadena sea impar, para este caso igual se tendrá que iterar las veces iguales a la mitad de la longitud haciendo uso de las reglas 4 y 5, pero al final como se necesita ser impar se tendrá que hacer uso ya sea de la regla 2 o la regla 3, para así conseguir una cadena impar.

El manejo de las cadenas fue prácticamente sencillo usando python y la función find() para ubicar siempre donde se encontraba a P para poder sustituirla por lo que se necesitara.

3. Desarrollo

3.1. Código del programa

Código creado con Python en el IDE de Pycharm:

```
1  # Programa 7 - palindromes
2  # Programa que construya palindromes de un lenguaje binario
3  # ENTRADA: Longitud del palindromo a generar (MAX 100,000)
4  # SALIDA: Un archivo de texto especificando
5  #         regla_usada      cadena_resultante      hasta llegar a la
        cadena final
6  # Debe tener opción manual y opción automática
7  """
8      REGLAS DE PRODUCCIÓN PARA
9      GRAMÁTICA LIBRE DE CONTEXTO
10     QUE CONSTRUYE PALINDROMES
11     (1) P → e
12     (2) P → 0
13     (3) P → 1
14     (4) P → 0P0
15     (5) P → 1P1
16 """
17 import random
18 import os
```

```

19
20 def cincuenta_cincuenta():
21     numero = random.randint(0, 1)
22     return numero
23
24 def sustituircaracter(conta, cadena, letra):
25     lista = list(cadena)
26     lista[conta] = letra
27     cadena = ''.join(lista)
28     return cadena
29
30 def sustituir_p(cadena, regla):
31     if regla == 1:
32         lugar = cadena.find('P')
33         i = 0
34         izquierda = ""
35         while i < lugar:
36             izquierda = izquierda + cadena[i]
37             i += 1
38         # print(izquierda)
39         j = lugar + 1
40         derecha = ""
41         while j < len(cadena):
42             derecha = derecha + cadena[j]
43             j += 1
44         # print(derecha)
45         cadena = izquierda + derecha
46         # print(cadena)
47         return cadena
48     elif regla == 2:
49         lugar = cadena.find('P')
50         cadena = sustituircaracter(lugar, cadena, '0')
51         # print(cadena)
52         return cadena
53     elif regla == 3:
54         lugar = cadena.find('P')
55         cadena = sustituircaracter(lugar, cadena, '1')
56         # print(cadena)
57         return cadena
58     elif regla == 4:
59         lugar = cadena.find('P')
60         i = 0
61         izquierda = ""
62         while i < lugar:
63             izquierda = izquierda + cadena[i]

```

```

64         i +=1
65         #print(izquierda)
66         j = lugar + 1
67         derecha = ""
68         while j < len(cadena):
69             derecha = derecha + cadena[j]
70             j += 1
71         #print(derecha)
72         cadena = izquierda + "0P0" + derecha
73         # print(cadena)
74         return cadena
75     elif regla == 5:
76         lugar = cadena.find('P')
77         i = 0
78         izquierda = ""
79         while i < lugar:
80             izquierda = izquierda + cadena[i]
81             i += 1
82         # print(izquierda)
83         j = lugar + 1
84         derecha = ""
85         while j < len(cadena):
86             derecha = derecha + cadena[j]
87             j += 1
88         # print(derecha)
89         cadena = izquierda + "1P1" + derecha
90         #print(cadena)
91         return cadena
92
93
94 def tituloarchivo():
95     archivo = open("Historia.txt", "a")
96     linea = "(REGLA)-> CADENA"
97     archivo.write(linea)
98     archivo.write('\n')
99     archivo.close()
100
101
102 def historia(cadena, regla):
103     archivo = open('Historia.txt', 'a')
104     linea = '(' + str(regla) + ')->' + cadena
105     archivo.write(linea)
106     archivo.write('\n')
107     archivo.close()
108

```

```

109 def crear_palindrome(longitud):
110     # Si es par -> iteraciones = n/2 + e al final (regla 1)
111     # Si es impar -> iteraciones = n/2 + alguna regla de 2 o 3
112     cadena = ""
113     conta = 0
114     mitad = longitud / 2
115     if longitud % 2 == 0: #par
116         while conta < mitad:
117             decision = cincuenta_cincuenta()
118             if decision == 0:
119                 #Regla 4
120                 cadena = sustitur_p(cadena, 4)
121                 historia(cadena, 4)
122             else:
123                 #Regla 5
124                 cadena = sustitur_p(cadena, 5)
125                 historia(cadena, 5)
126             #print(cadena)
127             conta += 1
128         cadena = sustitur_p(cadena, 1)
129         historia(cadena, 1)
130         #print(cadena)
131         return(cadena)
132     else: #impar
133         if longitud == 1:
134             uno = cincuenta_cincuenta()
135             if uno == 0:
136                 # Regla 2
137                 cadena = '0'
138                 historia(cadena, 2)
139             else:
140                 # Regla 3
141                 cadena = '1'
142                 historia(cadena, 3)
143         else:
144             while conta < int(mitad):
145                 decision = cincuenta_cincuenta()
146                 if decision == 0:
147                     #Regla 4
148                     cadena = sustitur_p(cadena, 4)
149                     historia(cadena, 4)
150                 else:
151                     #Regla 5
152                     cadena = sustitur_p(cadena, 5)
153                     historia(cadena, 5)

```

```

154         #print(cadena)
155         conta += 1
156         decision_final = cincuenta_cincuenta()
157         if decision_final == 0:
158             # Regla 2
159             cadena = sustitur_p(cadena, 2)
160             historia(cadena, 2)
161         else:
162             # Regla 3
163             cadena = sustitur_p(cadena, 3)
164             historia(cadena, 3)
165         #print(cadena)
166         return cadena
167
168
169 def limpiarpantalla():
170     if os.name == "posix":
171         os.system("clear")
172     elif os.name == "ce" or os.name == "nt" or os.name == "dos":
173         os.system("cls")
174
175 def manual():
176     limpiarpantalla()
177     print('Manual')
178     tituloarchivo()
179     try:
180         longitud = input(' De qu longitud se requiere el palindrome
181                             ?\n')
182         palindrome = crear_palindrome(int(longitud))
183         print('Palindrome creado:')
184         print(palindrome)
185         print('Archivo Historia Creado')
186         input()
187     except ValueError:
188         print('Ningun Valor recibido, intentar nuevamente')
189         input()
190
191 def automatico():
192     limpiarpantalla()
193     print('Automatico')
194     tituloarchivo()
195     longitud = random.randint(0, 100000)
196     print('El palindrome tendra una longitud de:')
197     print(longitud)
198     palindrome = crear_palindrome(longitud)

```

```

198     print('Palindrome creado:')
199     print(palindrome)
200     print('Archivo Historia Creado')
201     input()
202
203
204 def menu():
205     if os.path.isfile("Historia.txt"):
206         os.remove("Historia.txt")
207     while True:
208         limpiarpantalla()
209         print('Menu programa 7')
210         print('1.Forma Manual')
211         print('2.Forma Automatica')
212         print('3.Salir')
213         respuesta = input('Favor de escoger una opcion \n')
214         if respuesta == '1':
215             if os.path.isfile("Historia.txt"):
216                 os.remove("Historia.txt")
217             manual()
218         elif respuesta == '2':
219             if os.path.isfile("Historia.txt"):
220                 os.remove("Historia.txt")
221             automatico()
222         elif respuesta == '3':
223             break
224         else:
225             print('Ingrese un valore entre 1 y 3')
226             input()
227     print('Fin del programa')
228
229
230 if __name__ == '__main__':
231     menu()

```

3.2. Explicación

Funciones usadas en el programa:

cincuenta_cincuenta():

Esta función regresa el valor de la función random entre 0 y 1, que en si tienen una probabilidad del 50 % de ocurrir en ambos casos, esto sirve para la toma de decisiones aleatorias en las demás funciones.

sustituircaracter(conta, cadena, letra):

Esta función se encarga de sustituir un carácter dado, dentro de una cadena dada, en la posición especificada.

sustituir_p(cadena, regla):

Esta es la función que se encarga de sustituir a P dentro de las cadenas proporcionadas, y de acuerdo a la regla solicitada, dependiendo cual de las 5 reglas, se hacen cosas diferentes. En la regla 1 solo se elimina la P de la cadena dada para regresar una nueva cadena pero sin P. En la regla 2 y 3 se cambia la P por 0 o 1 dependiendo del caso. En la regla 4 y 5 se hace el cambio de P por 0P0 o 1P1 dependiendo el caso igualmente.

tituloarchivo():

Esta función pone la primer linea del archivo Historia.txt el cual menciona la estructura de los datos que después se podrán ver una vez creado un palindrome: (REGLA)->CADENA

historia(cadena, regla):

Es la función que se encarga de insertar en el archivo Historia.txt cada iteración en la cadena para que se convierta en el palindrome solicitado, se ingresa que regla se uso y la cadena actual.

crear_palindrome(longitud):

Es la función encargada de usar todas las demás funciones, así como varios ifs para crear los palindromes solicitados, de manera aleatoria, separando en casos dependiendo si se necesita ser par o impar, las reglas cambian ya sea el caso.

limpiarpantalla():

Es la función que limpia la consola/pantalla cuando se necesite.

manual():

Es la función encargada de solicitar al usuario la cantidad deseada para la creación del palindrome, una vez teniendo esto se llama a la función de crear_palindrome haciendo todo el proceso de creación del archivo Historia.txt y el palindrome con la longitud solicitada.

automatico():

Es la función encargada de obtener la longitud para el palindrome de manera aleatoria con un limite de 100,000 caracteres, una vez teniendo esto se llama a la función de crear_palindrome haciendo todo el proceso de creación del archivo Historia.txt y el palindrome con la longitud calculada.

menu():

Es la función que despliega el menú del programa con las opciones de forma manual, forma automática, y salir del programa.

3.3. Ejecución del programa, en Python

Ejecución del programa por consola de Pycharm:

```
Terminal: Local x + v
cesar@cesar-HP-Notebook:~/Documentos/Teoria de la Computación/Programa_7/python$ python3 main.py
```

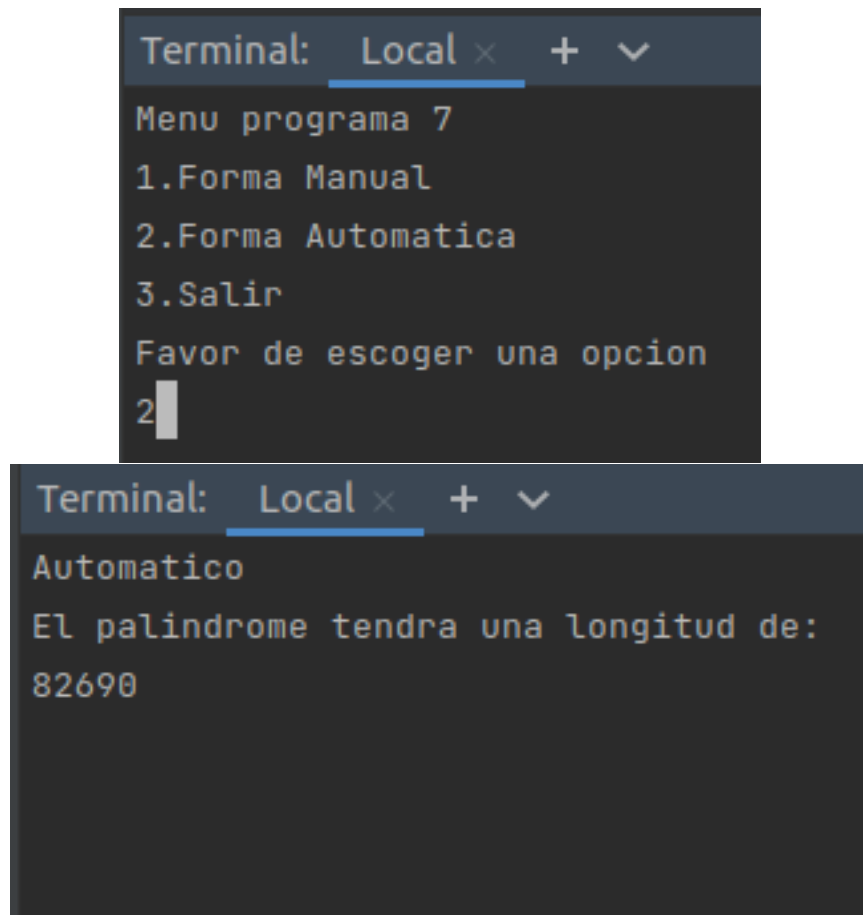
Menú del programa:

```
Terminal: Local x + v
Menu programa 7
1.Forma Manual
2.Forma Automatica
3.Salir
Favor de escoger una opcion
```

Opción 1. Forma Manual:

```
Terminal: Local x + v
Menu programa 7
1.Forma Manual
2.Forma Automatica
3.Salir
Favor de escoger una opcion
1
```


Opción 2. Forma Automática:

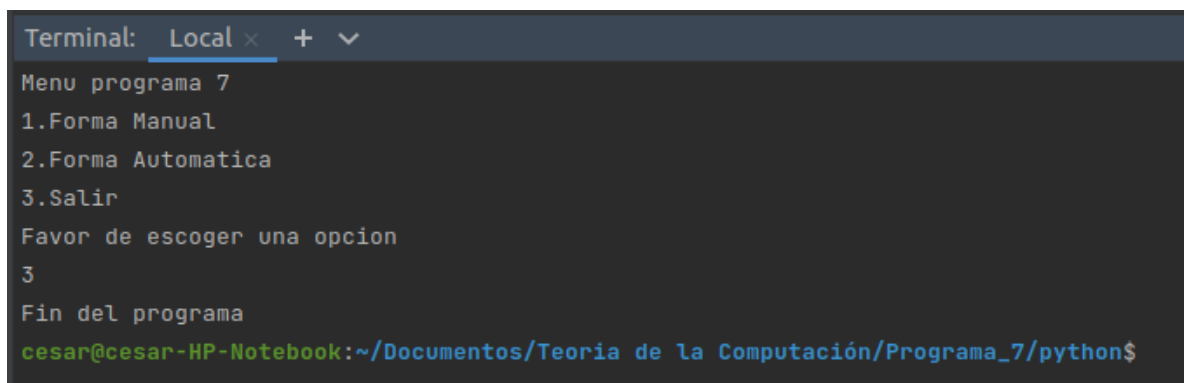


```
Terminal: Local x + v
Menu programa 7
1.Forma Manual
2.Forma Automatica
3.Salir
Favor de escoger una opcion
2

Terminal: Local x + v
Automatico
El palindrome tendra una longitud de:
82690
```

No se termino de crear el palindrome pues era de un tamaño demasiado grande, aunque en el rango de los 100,00. Si se le dejaba todo el tiempo que se necesitara, el archivo también quedaría demasiado grande.

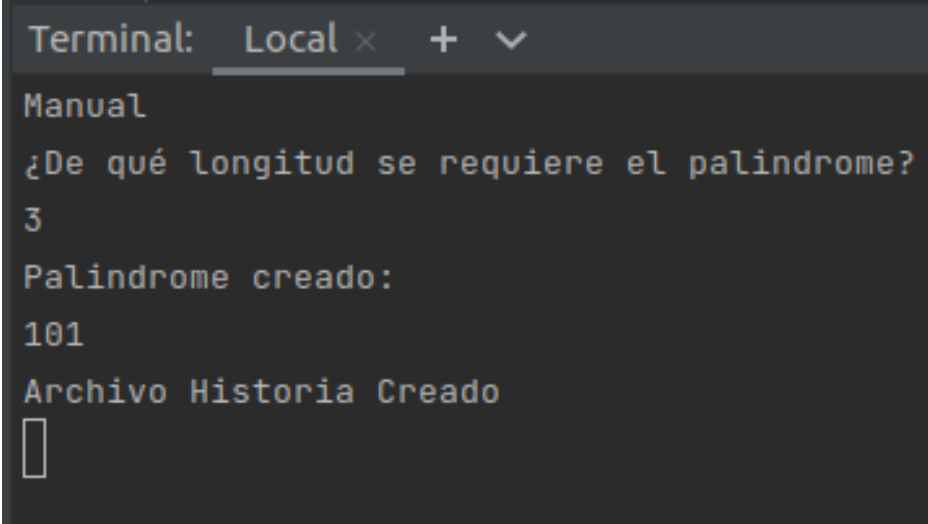
Opción 3. Salir del programa:



```
Terminal: Local x + v
Menu programa 7
1.Forma Manual
2.Forma Automatica
3.Salir
Favor de escoger una opcion
3
Fin del programa
cesar@cesar-HP-Notebook:~/Documentos/Teoria de la Computación/Programa_7/python$
```

4. Conclusiones

Este programa fue practicamente sencillo encontrar la solución para crear los palindromes pues solo se tenian que manejar cadenas, lo que ya habia echo antes por lo que no se me complico. Me gusto que se pudiera crear todo el archivo de Historia para cualquier palindrome creado, y que cada vez que se corriera otra vez aunque se requiriera la misma longitud, el palindrome siempre es diferente por las deciciones tomadas en cada vuelta. Un ejemplo sencillo de que el programa funciona es haciendo un palindrome de 3 de longitud:

A screenshot of a terminal window with a dark background. The title bar at the top says "Terminal: Local" followed by a close button (x), a plus sign (+), and a dropdown arrow (v). The terminal content shows the following text: "Manual" on the first line, "¿De qué longitud se requiere el palindrome?" on the second line, "3" on the third line, "Palindrome creado:" on the fourth line, "101" on the fifth line, and "Archivo Historia Creado" on the sixth line. A white cursor is visible on the seventh line.

```
Terminal: Local x + v
Manual
¿De qué longitud se requiere el palindrome?
3
Palindrome creado:
101
Archivo Historia Creado
█
```