

Práctica 5. Programa - palindromo

Autor: Victor Ulises Miranda Chávez. Grupo: 5BM1

Fecha de entrega: 06/11/2022

1. Introducción

Un palíndromo es una cadena de caracteres que se lee de la misma manera tanto de izquierda a derecha como de derecha a izquierda. Por ejemplo, "radar" es un palíndromo, ya que se lee igual en ambos sentidos.

En este programa se utiliza una gramática libre de contexto para construir palíndromos de un lenguaje binario, es decir, palíndromos que sólo contienen los símbolos 0 y 1.

1.1. Objetivo

Realizar un programa que construya palíndromos de un lenguaje binario. El lenguaje deberá solicitar únicamente la longitud del palíndromo a calcular, de esta manera el programa deberá construir el palíndromo de manera aleatoria. La longitud máxima que podría alcanzar un palíndromo será de 100,000 caracteres. La salida del programa se irá a un archivo de texto y ahí especificarán qué regla se seleccionó y la cadena resultante hasta llegar a la cadena final. El programa deberá ofrecer dos opciones: que el usuario defina la longitud del palíndromo o que lo genere todo de manera automática.

La gramática libre de contexto que construye palíndromos, se define con las siguientes reglas de producción.

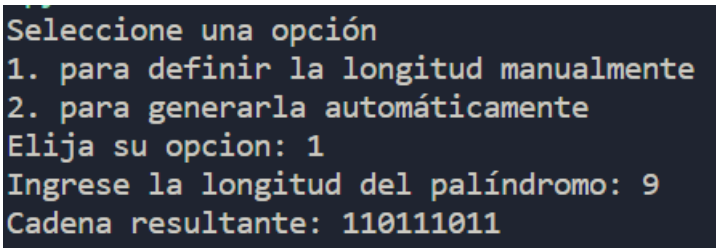
- (1) $P \rightarrow e$
- (2) $P \rightarrow 0$
- (3) $P \rightarrow 1$
- (4) $P \rightarrow 0P0$
- (5) $P \rightarrow 1P1$

2. Desarrollo

2.1. Lenguaje utilizado

- Python

2.2. Capturas de resultados:



```
Seleccione una opción
1. para definir la longitud manualmente
2. para generarla automáticamente
Elija su opción: 1
Ingrese la longitud del palíndromo: 9
Cadena resultante: 110111011
```

Figura 1: Selección del modo e ingreso de longitud

```
palindromo.txt
You, hace 1 segundo | 1 author (You)
1 Regla seleccionada: P -> 1P1
2 Regla seleccionada: P -> 1P1
3 Regla seleccionada: P -> 0P0
4 Regla seleccionada: P -> 1P1
5 Regla seleccionada: P -> 1
6 Cadena resultante: 110111011
```

Figura 2: Reglas seleccionadas

3. Conclusiones:

Desarrollar esta práctica fue retroalimentativa porque pude conocer la gramática de libre contexto basados en la construcción de palíndromos de un lenguaje binario. Por la parte práctica, fue bueno para fortalecer mis conocimientos en recursividad para poder generar un palíndromo de longitud n .

4. Bibliografía:

Hopcroft, J. E., Motwani, R. & Ullman, J. D. (2008). Teoría de autómatas, lenguajes y computación 3/E. Pearson Educación.

5. Código:

```
1 import random
2
3 def construir_palindromo(longitud):
4     f.write("Regla seleccionada: P -> ")
5     if longitud <= 0:
6         f.write("e\n")
7         return ""
8     elif longitud == 1:
9         c = random.choice(["0", "1"])
10        if c == "0":
11            f.write("0\n")
12        else:
13            f.write("1\n")
14        return c
15    elif longitud == 2:
16        c1 = random.choice(["0", "1"])
17        if c1 == "0":
18            f.write("0\n")
19        else:
20            f.write("1\n")
21
22    f.write("Regla seleccionada: P -> ")
23    if c1 == "0":
24        f.write("0\n")
25    else:
26        f.write("1\n")
27
28    return c1 + c1
29 else:
30     mitad = longitud - 2
31     caracter = random.choice(["0", "1"])
```

```

32         if character == "0" :
33             f.write("OP0\n")
34         else:
35             f.write("1P1\n")
36         return character + construir_palindromo(mitad) + character
37
38 # Solicitamos al usuario que seleccione una opción
39 opcion = input("Seleccione una opción\n1. para definir la longitud manualmente
40               \n2. para generarla automáticamente\nElija su opción: ")
41
42 if opcion == "1":
43     # El usuario define la longitud del palíndromo
44     longitud = int(input("Ingrese la longitud del palíndromo: "))
45 elif opcion == "2":
46     # Generamos una longitud aleatoria entre 1 y 100,000
47     longitud = random.randint(1, 100000)
48
49 f = open("palindromo.txt", "w")
50
51 # Construimos el palíndromo
52 palindromo = construir_palindromo(longitud)
53
54 f.write("Cadena resultante: " + palindromo + "\n")
55 print("Cadena resultante: " + palindromo + "\n")
56
57 # Cerramos el archivo
58 f.close()

```