Reporte de Algoritmos

Durante este trabajo estaré mostrando el análisis de algunos algoritmos, como el de pila, fila y grafos y los métodos para explorar en ellos.

Así mismo también mostrare los resultados obtenidos al hacer los algoritmos mencionados así como también cómo funcionan cada uno de ellos.

Es importante mencionar para que nos puede servir, ser útil vaya el análisis de algoritmos que gracias a estos se puede mejor estimación al tiempo así como es espacio de los recursos que se emplean.

Ahora hablare son uno de estos algoritmos de los que tanto he hablado que es fila

Fila: Su definición es sencilla, un conjunto de datos ordenados y se puede explorar en la fila a través del método BFS que nos permite saber su amplitud.

También estuvimos viendo otro análisis de algoritmo que es muy similar al de fila, que es pila solo que la diferencia es que es un conjunto de datos NO ordenados, además se puede explorar a través del método DFS para la profundidad.

En cuanto a la programación, nos permiten realizar la búsqueda de elementos dentro de un arreglo, permitiéndonos a su vez conocer la longitud del recorrido de cada uno de ellos

Se mostrara los Resultados de Fila y Pila

```
class fila:
    def __init__(self):
        self.fila=[ ]
    def obtener(self):
        return self.fila.pop()
    def meter(self,e):
        self.fila.insert(0,e)
        return len(self,fila)
    @property
    def longitud(self):
        return len(self.fila)
f=fila()
f.meter(1)
1
f.meter(2)
2
f.meter(100)
3
print(f.longitud)
3
```

Pila

```
class pila:
    def _init_(self):
        self.pila=[]
    def obtener(self):
        return self.pila.pop()
    def meter(self,e):
        self.pila.insert(0,e)
        return len(self.pila)
    @property
    def longitud(self):
        return len(self.pila)
p= pila()
p.meter(1)
```

1

```
p.meter(2)
```

2

```
p.meter(100)
```

3

```
p.obtener()
```

1

p.longitud

2

Hablare brevemente sobre grafo investigue un poco en Internet ya que el dia que el maestro nos explicó ese tema falte a clases así que por lo que encontré en internet encontré lo siguiente

Es un conjunto de objetos llamados vértices o nodos unidos por enlaces llamados aristas o arcos, que permiten representar relaciones binarias entre elementos de un conjunto. Un ejemplo de ello puede ser una red de computadoras puede representarse y estudiarse mediante un grafo, en el cual los vértices representan terminales y las aristas representan conexiones (las cuales, a su vez, pueden ser cables o conexiones inalámbricas).

El reultado fue el siguiente

```
class Grafo:
    def __init__(self):
        self.V=set()
        self.E=dict()
        self.vecinos=dict()
    def agrega(self,v):
        self.V.add(v)
        if not v in self.vecinos:
            self.vecinos[v]=set()
    def conecta(self,v,u,peso=1):
        self.agrega(v)
        self.agrega(u)
        self.E[(v,u)]=self.E[(u,v)]=peso
```

```
        self.vecinos[v].add(u)
        self.vecinos[u].add(v)
```

BFS Y DFS

Al igual que fila que fila y pila son casi similares como se pueden mostrar a continuación

BFS

```
def BFS(g,ni):
    visitados=[]
    f=fila()
    f.meter(ni)
    while(f.longitud>0):
        na=f.obtener()
        visitados.append(na)
        ln=G.vecinos[na]
        for nodo in ln:
            if nodo not in visitados:
                f.meter(nodo)
                return visitados
```

Mientras que el DFS:

```
def DFS(g,ni):
    f=pila()
    f.meter(ni)
    while(f.longitud>0):
        na=f.obtener()
        visitados.append(na)
        ln=G.vecinos[na]
        for nodo in ln:
            if nodo not in visitados:
                f.meter(nodo)
                return visitados
```

Conclusión

Este trabajo me pareció muy interesante y aprendi algo nuevo que aunque digan que esto para que nos puede servir como matemático, pero en realidad considero que si nos puede ser de gran ayuda ya que en los grafos se puede usar para modelar además tiene nodos y aristas puede estar orientado o no y puede ser planear o no planear.

En fin me gustaría seguir aprendiendo mas sobre todo esto