



UNIVERSIDAD JUÁREZ AUTÓNOMA DE TABASCO



División Académica De Ingeniería y Arquitectura

LICENCIATURA:

INGENIERIA ELECTRICA Y ELECTRONICA

ASIGNATURA:

ESTRUCTURA Y ALGORITMO DE DATOS

ACTIVIDAD:

2DO BLOQUE DE EJERCICIOS DE COLAS Y PILAS

DOCENTE:

PROF. MARIA ELENA GARCIA ULIN

ALUMNO QUE ENTREGA:

192D24008 IVÁN GARCÍA MÉNDEZ

GRUPO:

E12

EJERCICIOS DE PILAS Y COLAS

EJERCICIO 1

1. Escribir una clase TorreDeControl que modele el trabajo de una torre de control de un aeropuerto, con una pista de aterrizaje. La torre trabaja en dos etapas: reconocimiento y acción.

```
G:\6TO SEMESTRE IEE\ALGORITMO Y ESTRUCTURA DE DATOS\EJ_1.1.py
EJ_1.1.py X EJ_1.3.py X EJ_1.4.py X EJ_2.1.py X EJ_2.2.py X EJ_2.3.py X
4
5 @author: CrIva
6 """
7
8 class TorreDeControl:
9
10     def __init__(self):
11         self.Despegues=["AV-03","AV-04","AV-06","AV-08","AV-010"]
12         self.Aterrizajes=["AV-01","AV-05","AV-09","AV-02","AV-07"]
13         self.Estacionamiento1=["ES1","ES2","ES3","ES4","ES5","ES6","ES7","ES8"]
14
15     def Accion(self,A1):
16         self.Despegues.remove(A1)
17         self.Aterrizajes.append(A1)
18     def Aterrizajes(self,A3):
19         self.Aterrizajes.remove(A3)
20         self.Despegues.append(A3)
21     def Reconocimiento(self,R1):
22         self.Aterrizajes.append(R1)
23
24     def Estacionamiento(self):
25         print(self.Estacionamiento1)
26
27     def mostrar(self):
28         print("Aviones por aterrizar")
29         print(self.Aterrizajes,sep=" ")
30         print("Aviones que despegan")
31         print(self.Despegues,sep=" ")
32
33 Torre=TorreDeControl()
34 while True:
35     print("1-Asignar estacionamiento para aviones recién llegados")
36     print("2-Ingresa aviones que despegan")
37     print("3-Reconocimiento de aviones por despegaron y aterrizaron")
38     print("4-Salir")
39     op=int(input("opcion:"))
40     if op==1:
41         print("Estacionamiento disponibles")
42         print(Torre.Estacionamiento())
43         p=int(input("si quiere ocupar un lugar presione 2, si no presione 1:"))
44         if p==2:
45             T=str(input("Ingrese el avion que aterrizo:"))
46             Torre.Aterrizajes.remove(T)
47             Torre.Despegues.append(T)
48             Lugar=str(input("Ingrese el estacionamiento a asignar:"))
49             Torre.Estacionamiento1.remove(Lugar)
50         if p==1:
51             Des=str(input("Ingrese el estacionamiento a desocupar:"))
52             Torre.Estacionamiento1.append(Des)
53     elif op==2:
54         Des=str(input("Ingrese el avion que despegó:"))
55         Torre.Accion(Des)
56     elif op==3:
57         print(Torre.mostrar())
58     elif op==4:
59         break
60     else :
61         print("Instruccion invalida")
```

Captura de pantalla del código correspondiente al problema 1.1

```
self.Despegues=["AV-03","AV-04","AV-06","AV-08","AV-010"]
self.Aterrizajes=["AV-01","AV-05","AV-09","AV-02","AV-07"]
self.Estacionamiento1=["ES1","ES2","ES3","ES4","ES5","ES6","ES7","ES8"]
```

Cola de aviones, aterrizajes y estacionamientos

```
Console 1/A X
opcion:4
In [20]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_1.1.py',
wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Asignar estacionamiento para aviones recién llegados
2-Ingresa aviones que despegan
3-Reconocimiento de aviones que despegaron y aterrizaron
4-Salir

opcion:1
Estacionamiento disponibles
['ES1', 'ES2', 'ES3', 'ES4', 'ES5', 'ES6', 'ES7', 'ES8']
None

si quiere ocupar un lugar presione 2, si no presione 1:1

Ingrese el estacionamiento a desocupar:ES1
1-Asignar estacionamiento para aviones recién llegados
2-Ingresa aviones que despegan
3-Reconocimiento de aviones que despegaron y aterrizaron
4-Salir

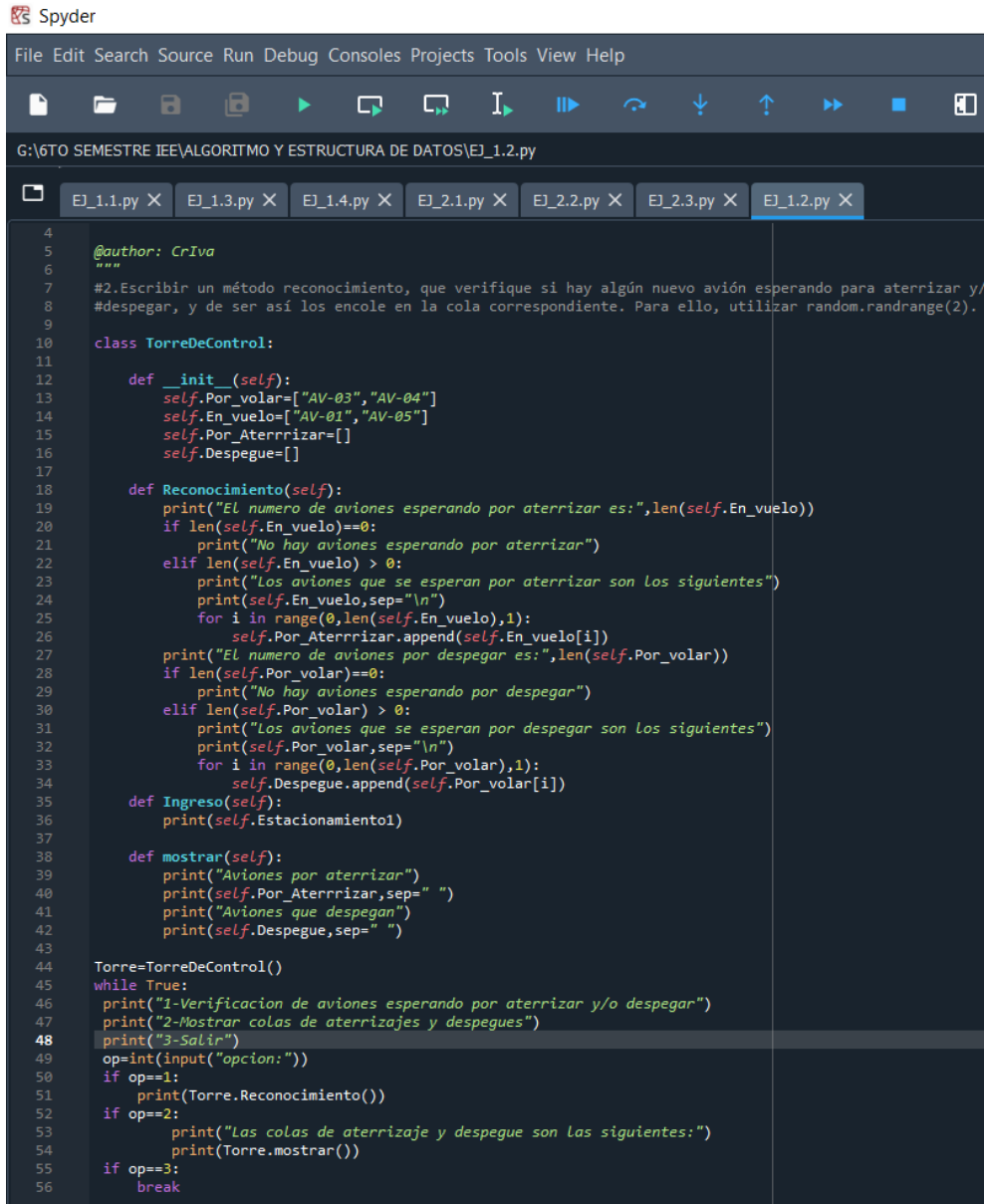
opcion:2

Ingrese el avion que despegue:AV-03
1-Asignar estacionamiento para aviones recién llegados
2-Ingresa aviones que despegan
3-Reconocimiento de aviones que despegaron y aterrizaron
4-Salir

opcion:3
Aviones por aterrizar
['AV-01', 'AV-05', 'AV-09', 'AV-02', 'AV-07', 'AV-03']
.
.
Aviones que despegan
['AV-04', 'AV-06', 'AV-08', 'AV-010']
```

Ventana de resultado, en este caso muestra que los estaciones que están disponibles, y reconocen los aviones que están despegando y aterrizando

2. Escribir un método reconocimiento, que verifique si hay algún nuevo avión esperando para aterrizar y/o despegar, y de ser así los encole en la cola correspondiente.



```
4
5  @author: CrIva
6  """
7  #2.Escribir un método reconocimiento, que verifique si hay algún nuevo avión esperando para aterrizar y/o
8  #despegar, y de ser así los encole en la cola correspondiente. Para ello, utilizar random.randrange(2).
9
10 class TorreDeControl:
11
12     def __init__(self):
13         self.Por_volar=["AV-03","AV-04"]
14         self.En_vuelo=["AV-01","AV-05"]
15         self.Por_Aterrrizar=[]
16         self.Despegue=[]
17
18     def Reconocimiento(self):
19         print("El numero de aviones esperando por aterrizar es:",len(self.En_vuelo))
20         if len(self.En_vuelo)==0:
21             print("No hay aviones esperando por aterrizar")
22         elif len(self.En_vuelo) > 0:
23             print("Los aviones que se esperan por aterrizar son los siguientes")
24             print(self.En_vuelo,sep="\n")
25             for i in range(0,len(self.En_vuelo),1):
26                 self.Por_Aterrrizar.append(self.En_vuelo[i])
27         print("El numero de aviones por despegar es:",len(self.Por_volar))
28         if len(self.Por_volar)==0:
29             print("No hay aviones esperando por despegar")
30         elif len(self.Por_volar) > 0:
31             print("Los aviones que se esperan por despegar son los siguientes")
32             print(self.Por_volar,sep="\n")
33             for i in range(0,len(self.Por_volar),1):
34                 self.Despegue.append(self.Por_volar[i])
35     def Ingreso(self):
36         print(self.Estacionamiento1)
37
38     def mostrar(self):
39         print("Aviones por aterrizar")
40         print(self.Por_Aterrrizar,sep=" ")
41         print("Aviones que despegan")
42         print(self.Despegue,sep=" ")
43
44 Torre=TorreDeControl()
45 while True:
46     print("1-Verificacion de aviones esperando por aterrizar y/o despegar")
47     print("2-Mostrar colas de aterrizajes y despegues")
48     print("3-Salir")
49     op=int(input("opcion:"))
50     if op==1:
51         print(Torre.Reconocimiento())
52     if op==2:
53         print("Las colas de aterrizaje y despegue son las siguientes:")
54         print(Torre.mostrar())
55     if op==3:
56         break
```

Captura de pantalla del código correspondiente al problema 1.2

1-Verificacion de aviones esperando por aterrizar y/o despegar
2-Mostrar colas de aterrizajes y despegues
3-Salir

Este es el menú en el cual si se presiona **1** verifica los aviones esperando por aterrizar y/o despegar que es lo que pide el problema

```
In [2]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_1.2.py',
wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Verificacion de aviones esperando por aterrizar y/o despegar
2-Mostrar colas de aterrizajes y despegues
3-Salir

opcion:1
El numero de aviones esperando por aterrizar es: 2
Los aviones que se esperan por aterrizar son los siguientes
['AV-01', 'AV-05']
El numero de aviones por despegar es: 2
Los aviones que se esperan por despegar son los siguientes
['AV-03', 'AV-04']
None
```

Ventana de resultado, en este caso muestra que la cola que esta **En_vuelo** ya que se esperan que aterricen, los aviones que están por despegar son los que se encuentran en la cola **Por_volar**.

3. Escribir un método acción, que haga aterrizar o bien despegar, al primero de los aviones que este esperando (los que esperan para aterrizar tienen prioridad). Debe desencolar el avión de su cola y devolver la información correspondiente.

```
G:\6TO SEMESTRE IEE\ALGORITMO Y ESTRUCTURA DE DATOS\EJ_1.3.py
EJ_1.1.py X EJ_1.3.py X
4
5 @author: CrIva
6 """
7
8 #Escribir un método acción, que haga aterrizar o bien despegar, al primero de los aviones que este esperando
9 # (los que esperan para aterrizar tienen prioridad). Debe desencolar el avión de su cola y devolver la información
10 # correspondiente.
11
12 class TorreDeControl:
13
14     def __init__(self):
15         self.Por_volar=["AV-03","AV-04","AV-02","AV-07"]
16         self.En_vuelo=["AV-01","AV-05","AV-08","AV-010"]
17         self.Por_Aterrrizar=[]
18
19     def Accion(self,n1):
20         if n1==1:
21             print("El avion que aterrizara es:")
22             self.Por_Aterrrizar.append(self.En_vuelo[0])
23             print(self.En_vuelo[0])
24             self.En_vuelo.remove(self.En_vuelo[0])
25
26         if n1==2:
27             print("El avion por despegar es")
28             print(self.Por_volar[0])
29             self.En_vuelo.append(self.Por_volar[0])
30
31     def __str__(self):
32         if len(self.Por_Aterrrizar)==0:
33             print("No hay aviones en la cola para aterrizar")
34         if len(self.Por_Aterrrizar)>0:
35             print("Los Aviones que aterrizaron son:")
36             print(self.Por_Aterrrizar,sep=" ")
37         if len(self.En_vuelo)==0:
38             print("No hay aviones en la cola para despegar")
39         if len(self.En_vuelo)>0:
40             print("Los Aviones que despegaron son:")
41             print(self.Por_volar,sep=" ")
42
43 Torre=TorreDeControl()
44 while True:
45     print("1-Accion de despegar o aterrizar")
46     print("2-Mostrar estado actual colas de aterrizajes y despegues")
47     print("3-Salir")
48     op=int(input("opcion:"))
49     if op==1:
50         p=int(input("Teclee 1(aterrizar) o 2(despegar) segun la accion a elegir:"))
51         print(Torre.Accion(p))
52     if op==2:
53         print("Las colas de aterrizaje y despegue son las siguientes:")
54         print(Torre.__str__())
55     if op==3:
56         break
```

Captura de pantalla del código correspondiente al problema 1.3

Colas usar Como ejemplo:

```
self.Por_volar=["AV-03","AV-04","AV-02","AV-07"]
self.En_vuelo=["AV-01","AV-05","AV-08","AV-010"]
self.Por_Aterrrizar=[]
```

```
In [14]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_1.3.py',
wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Accion de despegar o aterrizar
2-Mostrar estado actual colas de aterrizajes y despegues
3-Salir

opcion:1

Teclee 1(aterrizar) o 2(despegar) segun la accion a elegir:1
El avion que aterrizara es:
AV-01
None
1-Accion de despegar o aterrizar
2-Mostrar estado actual colas de aterrizajes y despegues
3-Salir

opcion:1

Teclee 1(aterrizar) o 2(despegar) segun la accion a elegir:2
El avion por despegar es
AV-03
0
```

En este caso el programa muestra el avión que va a despegar y aterrizar que es lo que pide el problema, en ejemplo

4. Escribir un método `__str__` que imprima el estado actual de ambas colas.

```
G:\6TO SEMESTRE IEE\ALGORITMO Y ESTRUCTURA DE DATOS\EJ_1.4.py
EJ_1.1.py X EJ_1.3.py X EJ_1.4.py X

1  # -*- coding: utf-8 -*-
2  """
3  Created on Sun Jun  5 13:11:04 2022
4
5  @author: CrIva
6  """
7
8  #Escribir un método acción, que haga aterrizar o bien despegar, al primero de los aviones que este esperando
9  # (los que esperan para aterrizar tienen prioridad). Debe desencolar el avión de su cola y devolver la información
10 #correspondiente.
11
12 class TorreDeControl:
13
14     def __init__(self):
15         self.Por_volar=["AV-03","AV-04","AV-02","AV-07"]
16         self.En_vuelo=["AV-01","AV-05","AV-08","AV-010"]
17         self.Por_Aterrizar=[]
18
19     def Accion(self,n1):
20         if n1==1:
21             print("El avion que aterrizar es:")
22             self.Por_Aterrizar.append(self.En_vuelo[0])
23             print(self.En_vuelo[0])
24             self.En_vuelo.remove(self.En_vuelo[0])
25
26         if n1==2:
27             print("El avion por despegar es")
28             print(self.Por_volar[0])
29             self.En_vuelo.append(self.Por_volar[0])
30
31     def __str__(self):
32         if len(self.Por_Aterrizar)==0:
33             print("No hay aviones en la cola para aterrizar")
34         if len(self.Por_Aterrizar)>0:
35             print("Los Aviones que aterrizaron son:")
36             print(self.Por_Aterrizar,sep=" ")
37         if len(self.En_vuelo)==0:
38             print("No hay aviones en la cola para despegar")
39         if len(self.En_vuelo)>0:
40             print("Los Aviones que despegaron son:")
41             print(self.Por_volar,sep=" ")
42
43 Torre=TorreDeControl()
44 while True:
45     print("1-Accion de despegar o aterrizar")
46     print("2-Mostrar estado actual colas de aterrizajes y despegues")
47     print("3-Salir")
48     op=int(input("opcion:"))
49     if op==1:
50         p=int(input("Teclee 1(aterizar) o 2(despegar) segun la accion a elegir:"))
51         print(Torre.Accion(p))
52     if op==2:
53         print("Las colas de aterrizaje y despegue son las siguientes:")
54         print(Torre.__str__())
55     if op==3:
56         break
```

Captura de pantalla del código correspondiente al problema 1.4

```
In [13]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_1.4.py',
wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Accion de despegar o aterrizar
2-Mostrar estado actual colas de aterrizajes y despegues
3-Salir

opcion:1

Teclee 1(aterizar) o 2(despegar) segun la accion a elegir:1
El avion que aterrizar es:
AV-01
None
1-Accion de despegar o aterrizar
2-Mostrar estado actual colas de aterrizajes y despegues
3-Salir

opcion:1

Teclee 1(aterizar) o 2(despegar) segun la accion a elegir:2
El avion por despegar es
AV-03
None
1-Accion de despegar o aterrizar
2-Mostrar estado actual colas de aterrizajes y despegues
3-Salir

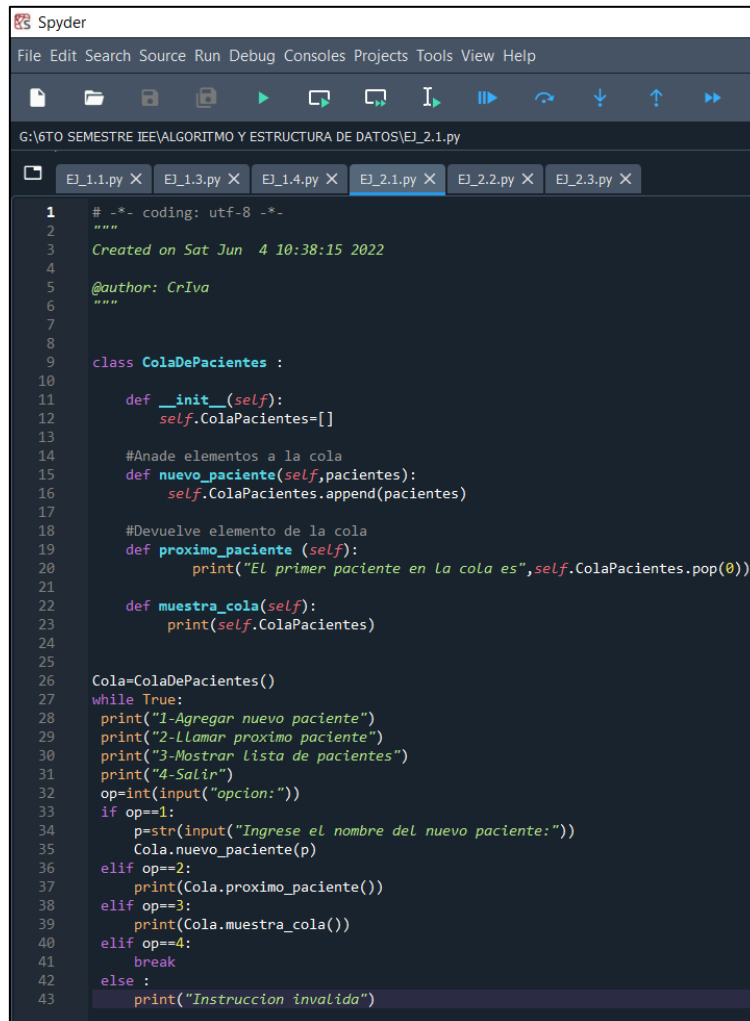
opcion:2
Las colas de aterrizaje y despegue son las siguientes:
Los Aviones que aterrizaron son:
['AV-01']
Los Aviones que despegaron son:
['AV-03', 'AV-04', 'AV-02', 'AV-07']
```

Ventana de resultado, en este caso el ejercicio nos pide que mostremos el estado de las colas, para este caso al oprimir la opción **2** muestra las colas de aterrizaje y la cola de despegue

EJERICICIO 2

Atención a los pacientes de un consultorio médico, con varios doctores

1. Escribir una clase ColaDePacientes con los métodos nuevo_paciente que reciba el nombre del paciente y lo encole, y un método proximo_paciente que devuelva el primer paciente en la cola y lo desencole.



```
1  # -*- coding: utf-8 -*-
2
3  Created on Sat Jun  4 10:38:15 2022
4
5  @author: CrIva
6
7
8
9  class ColaDePacientes :
10
11      def __init__(self):
12          self.ColaPacientes=[]
13
14      #Añade elementos a la cola
15      def nuevo_paciente(self,pacientes):
16          self.ColaPacientes.append(pacientes)
17
18      #Devuelve elemento de la cola
19      def proximo_paciente (self):
20          print("El primer paciente en la cola es",self.ColaPacientes.pop(0))
21
22      def muestraCola(self):
23          print(self.ColaPacientes)
24
25
26  Cola=ColaDePacientes()
27  while True:
28      print("1-Agregar nuevo paciente")
29      print("2-Llamar proximo paciente")
30      print("3-Mostrar lista de pacientes")
31      print("4-Salir")
32      op=int(input("opcion:"))
33      if op==1:
34          p=str(input("Ingrese el nombre del nuevo paciente:"))
35          Cola.nuevo_paciente(p)
36      elif op==2:
37          print(Cola.proximo_paciente())
38      elif op==3:
39          print(Cola.muestraCola())
40      elif op==4:
41          break
42      else :
43          print("Instruccion invalida")
```

Captura de pantalla del código correspondiente al problema 2.1

```
print("1-Agregar nuevo paciente")
print("2-Llamar proximo paciente")
print("3-Mostrar lista de pacientes")
print("4-Salir")
```

Este es el menú en el cual funciona el programa para que funcione se debe presionar el numero correspondiente **1 para agregar pacientes**, **2 para llamar el próximo paciente** y **3 para mostrar la cola de pacientes**


```

In [8]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_2.1.py',
            wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:1

Ingrese el nombre del nuevo paciente:Ana
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:1

Ingrese el nombre del nuevo paciente:Ivan
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:1

Ingrese el nombre del nuevo paciente:Oscar
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

Ingrese el nombre del nuevo paciente:Ivan
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:1

Ingrese el nombre del nuevo paciente:Oscar
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:3
['Ana', 'Ivan', 'Oscar']
None
1-Agregar nuevo paciente
2-Llamar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:2
El primer paciente en la cola es Ana

```

Ventana de resultado, en este caso el ejercicio nos pide ingresar pacientes esto en el programa se logra con la opción 1 como podemos ver en la imagen ingresamos 3 pacientes: primero **Ana**, luego **Ivan** y por último **Oscar**, con la opción 3 se muestra la cola de pacientes, con la opción 2 devuelve el primer paciente en la cola la cual en este caso es **Ana**.

2. Escribir una clase Recepción, que contenga un diccionario con las colas correspondientes a cada doctor o doctora, y los métodos nuevos _paciente que reciba el nombre del paciente y del especialista, y proximo_paciente que reciba el nombre de la persona liberada y devuelva el próximo paciente en espera

```
G:\6TO SEMESTRE IEE\ALGORITMO Y ESTRUCTURA DE DATOS\EJ_2.2.py
EJ_1.1.py X EJ_1.3.py X EJ_1.4.py X EJ_2.2.py X EJ_2.3.py X

1  # -*- coding: utf-8 -*-
2  """
3  Created on Sat Jun  4 15:11:03 2022
4
5  @author: CrIva
6  """
7
8  class Recepcion :
9
10     def __init__(self):
11         self.paciente=["Ana","Diego","Ivan"]
12         self.medic=["Dr.Rojas","Dr.Medina","Dr.Luis"]
13
14     def nuevos_pacientes(self,pacientes,especialista):
15         self.paciente.append(pacientes)
16         self.medic.append(especialista)
17     #Devuelve elemento de la cola
18     def proximo_paciente (self,pacientes):
19         if pacientes==self.paciente[0]:
20             print("El paciente liberad@ es",self.paciente.pop(0))
21             self.medic.pop(0)
22             return print("El proximo paciente es",self.paciente[0])
23             self.medic.pop(0)
24         if len(self.paciente) < 1:
25             print("")
26     def muestra_cola(self):
27         print(list(zip(self.paciente,self.medic)))
28
29 Recepcion=Recepcion()
30 while True:
31     print("1-Agregar nuevo paciente")
32     print("2-Ingresa persona liberada y llamar proximo paciente")
33     print("3-Mostrar lista de pacientes y doctor@s")
34     print("4-Salir")
35     op=int(input("opcion:"))
36     if op==1:
37         paciente=str(input("Nombre del paciente:"))
38         especialista=str(input("Nombre del Especialista:"))
39         Recepcion.nuevos_pacientes(paciente,especialista)
40     elif op==2:
41         pacientefree=str(input("Nombre del paciente liberado:"))
42         Recepcion.proximo_paciente(pacientefree)
43     elif op==3:
44         print(Recepcion.muestra_cola())
45     elif op==4:
46         break
47     else :
48         print("Instruccion invalida")
49
```

Captura de pantalla del código correspondiente al problema 2.2

```
In [9]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_2.2.py',
wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Agregar nuevo paciente
2-Ingresa persona liberada y llamar proximo paciente
3-Mostrar lista de pacientes y doctor@s
4-Salir

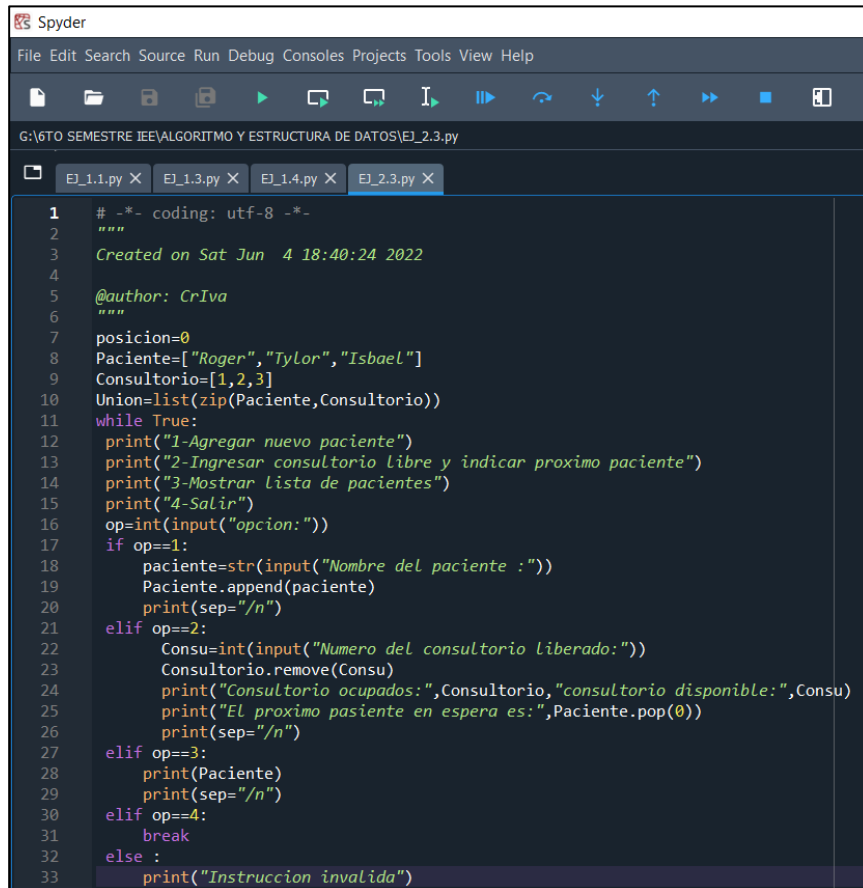
opcion:2

Nombre del paciente liberado:Ana
El paciente liberad@ es Ana
El proximo paciente es Diego
```

Ventana de resultado, en este caso muestra se muestra el paciente liberad@ y el paciente próximo teniendo en cuenta la siguiente cola de pacientes

```
self.paciente=["Ana","Diego","Ivan"]
self.medic=["Dr.Rojas","Dr.Medina","Dr.Luis"]
```

3. Escribir un programa que permita al usuario ingresar nuevos pacientes o indicar que un consultorio se ha liberado y en ese caso imprima el próximo



```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Sat Jun  4 18:40:24 2022
4
5 @author: CrIva
6 """
7 posicion=0
8 Paciente=["Roger","Tylor","Isbael"]
9 Consultorio=[1,2,3]
10 Union=list(zip(Paciente,Consultorio))
11 while True:
12     print("1-Agregar nuevo paciente")
13     print("2-Ingresa consultorio libre y indicar proximo paciente")
14     print("3-Mostrar lista de pacientes")
15     print("4-Salir")
16     op=int(input("opcion:"))
17     if op==1:
18         paciente=str(input("Nombre del paciente :"))
19         Paciente.append(paciente)
20         print(sep="/n")
21     elif op==2:
22         Consu=int(input("Numero del consultorio liberado:"))
23         Consultorio.remove(Consu)
24         print("Consultorio ocupados:",Consultorio,"consultorio disponible:",Consu)
25         print("El proximo pasiente en espera es:",Paciente.pop(0))
26         print(sep="/n")
27     elif op==3:
28         print(Paciente)
29         print(sep="/n")
30     elif op==4:
31         break
32     else :
33         print("Instruccion invalida")
```

Captura de pantalla del código correspondiente al problema 2.3

```
Paciente=["Roger","Tylor","Isbael"]
Consultorio=[1,2,3]
```

Ventana de resultado, en este caso se muestra el consultorio liberado y el paciente próximo

```
In [12]: runfile('G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS/EJ_2.3.py',
wdir='G:/6TO SEMESTRE IEE/ALGORITMO Y ESTRUCTURA DE DATOS')
1-Agregar nuevo paciente
2-Ingresa consultorio libre y indicar proximo paciente
3-Mostrar lista de pacientes
4-Salir

opcion:2

Numero del consultorio liberado:1
Consultorio ocupados: [2, 3] consultorio disponible: 1
El proximo pasiente en espera es: Roger
```