Tecnologico Nacional de Mexico
Instituto Tecnologico Tijuana
Subdireccion Academica
Departamento de Sistemas y Computacion
Ing. Sistemas Computacionales
Estructura de Datos

Profesor: Ray Brunett Parra Galaviz

Alumno: Delgado Vasquez Erik **Matricula: 17211515**

I. Leer detalladamente los ejercicios y codificar su respuesta en cualquier lenguaje de programacion

1. Realizar un programa que entregue el resultado de sumar 1+2+3+4+5+6+7+8+9, utilizando recursividad.

```
#Delgado Vasquez Erik No.Matricula 172115151
def suma(n):        #Se crea el metodo suma
   if (n==1):         #se crea la condicion
      return 1        #si el valor de n fuese "1" se regresaria el valor de "1"
   else:
      return n+suma(n-1)      #En caso contrario, se suma el valor de "n"
                              #con el valor anterior a el mismo, esto
n=9                           #mandando a llamar a su mismo metodo por medio de recursividad
print("La suma de 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 es de: "+ str(suma(n))) #se imprime la suma de los
numeros
```

2. Realizar un programa que entregue el resultado de la siguiente ecuacion 2^n, utilizar recursividad

```
#Delgado Vasquez Erik No.Matricula 172115151
import sys
global a

global n
def potencia(a,n): #se crea el metodo potencia que recibe dos parametros
   if (n ==0):    #se crea la condicion donde si el valor de "n" es "0" regresara como resultado "1"
      return 1

   else:
      return a * potencia(a, n-1)  #En caso contrario realizara la operacion debida para dar el resultado de
la funcion

global a
a=2

global n # se crea una variable glonal "n"
try:    #se crea un try para ingresar el valor del exponencial
   n=int(input("Ingrese el valor de la potencia: "))
except ValueError:  #si el termino ingresado no es un entero saltara la excepcion
   print("Ingrese un dato valido. ")
else:
   print("El resultado de la funcion es: " + str(potencia(a,n))) #si el valor ingresado es valido se imprime
el resultado


def repetir():          #se crea el metodo repetir para que el usuario elija si quiere realizar la funcion con
otro valor del exponente
   print("1.-Utilizar otro exponente. ")
   print("2.-Salir del programa. ")
   r=int(input("¿Qué desea hacer?. ")) #se pregunta al usuario su decision
   if r==1:
      a=2                  #si la opcion es la primera se repite el proceso de capturar el exponente e imprimir
el resutlado
      try:
         n=int(input("Ingrese el valor de la potencia: "))
      except ValueError:
         print("Ingrese un dato valido. ")
         repetir()
      else:
         print("El resultado de la funcion es: ", potencia(a,n))
         repetir()

   else:                   #en caso contrario se imprime el texto "Fin del programa" y termina el mismo.
      if r==2:
         print("Fin del programa.")
         sys.exit()
```

```
    else:
        print("Opción incorrecta, introduzca una opción válida.¨")      #Si el usuario ingresa una opcion
que no se encuentra en el menu salta
        input()                                          #la advertencia y le pide ingresar datos existentes
        repetir()

repetir()
```

3. Hacer un programa que lleve el control de versiones de un proyecto en una empresa, la estructura de datos debe controlar el numero de migraciones realizadas en el proyecto. Utilizar una pila para introducir las migraciones una por una y obtener las migraciones una por una empezando por la migracion mas actual y terminando por la migracion mas antigua.

```
#Delgado Vasquez Erik No.Matricula 172115151
import sys


class proyecto():  #se crea el metodo proyecto
   stack=[]       #se inicializa la pila

   def Crear(self):    #se crea el metodo crear, en el cual, como su nombre lo dice, se crea la pila
      self.stack=[]
      print("Proyecto creado exitosamente. ")
      input()

   def push(self):        #creacion del metodo push
      try:
         n=float(input("Ingrese numero de migracion: "))     #se pidel al usuario ingrese el numero de
migracion del proyecto
         self.stack.append(n)                        #Nota: con numero de migracion nos referimos a la
actualizacion del proyecto
         print("Migracion ingresada. ")                  #Ejemplo: Migracion 1.75, migracion 1.79, migracion
2.0, etc.
         input()

      except:
         print("No existe proyecto, favor de crearlo. ")     #se pone una excepcion, la cual nos pedira que se
cree un proyecto en caso de que
         input()                                    #el usuaio quiera ingresar migraciones sin existir un proyecto

   def peek(self):                           #creacion del metodo peek
      if 0 < len(self.stack):                  #condicion que nos permitira ingresar migraciones mientras la
longitud de la pila sea mayor que cero
         print(self.stack[len(self.stack)-1])     #se imprime la ultima migracion realizada
         self.stack.pop()                     #saca la migracion que se encuentra en el ultimo lugar, para
despues mostrar la migracion anterior a esta ultima
         input()
      else:
         print("No hay migraciones para mostrar. ") #En cso de que se hayan mostrado todas las
migraciones
                                          #se pone una advertencia de que ya no hay migraciones mas antiguas
y no se puede mostrar nada
   def size(self):                 #creacion del metodo size
      print(len(self.stack))         #se imprime la longitud de la pila, el numero de migraciones ingresadas

   def Migraciones(self):         #creacion del metodo migraciones
      print(self.stack)          #se muestran todas las migraciones que haya existentes en el proyecto
      print("Migraciones ingresadas del proyecto. ")
```

```
stack=proyecto() #se crea el objeto stack para ser utilizado posteriormente


while True:

   print("Menú")                              #creacion del menu para dar a elegir al usuario que desea hacer
   print("1.- Crear un proyecto. ")                #se debe crear obligatoriamente el proyecto para seguir
utilizando el programa
   print("2.- Introducir migracion. ")
   print("3.- Obtener migracion. ")
   print("4.- Saber numero de migraciones. ")
   print("5.- Mostras todas las migraciones. ")
   print("6.- Salir del programa. ")

   opc=int(input("¿Qué desea hacer? "))       #lectura de la opcion tomada por el usuario
   if opc==1:
      stack.Crear()          #se manda a llamar al metodo crear
   else:
      if opc==2:
         stack.push()       #se manda a llamar ak metodo push para ingresar migraciones al proyecto
      else:
         if opc==3:
            stack.peek()       #se manda a llamar al metodo peek apra mostrar la ultima migracion
         else:
            if opc==4:
               stack.size()       #se manda a llamar al metodo size para saber cuantas migraciones hay
existentes en el proyecto
            else:
               if opc==5:
                  stack.Migraciones()     #se manda a llamar al metodo migraciones para mostrar todas las
migraciones existentes en el proyecto
               else:
                  if opc==6:
                     print("Fin del programa. ")
                     sys.exit()               #se finaliza el programa
```

4. Hacer un programa que simule la fila de clientes de una tienda de super mercado, considerando que solo hay una caja que esta activa. La fila solo puede tener como maximo 5 clientes!

```
#Delgado Vasquez Erik No.Matricula 172115151
import sys
ind=0       #se inicializa la variable del indice
value='Empty'       #se inicializa la variable del valor de cada ezpacio en la fila/cola

def new():      #creacion del metodo new
    fila=['Empty','Empty','Empty','Empty','Empty']      #creacion de la cola llamada fila
    ind=0
    return fila,ind     #se regresan los valores del indice y la fila para ser utilizados posteriormente


queue, ind = new()  #se crea el objeto del metodo new para utilizar la cola y almacenar datos
print('*' * 25)     #separador
print(queue)        #se imprime la fila

print('*' * 25)

def formar(queue, value, ind):      #creacion del metodo formar/push que recibe parametros de la cola, sus valores y el indice
    if ind ==0:                     #condicion en la que mientras el indice sea igual a cero, se agregaran los datos en ese indice
        queue[ind]=value;
        ind+=1                      #incremento del indice
        print('*' * 25)
        print("El cliente se ha formado. ")
        print('*' * 25)
        return queue, value, ind        #se regresan los valores de los parametros
    elif ind ==1:                   #se repite el mismo procedimiento, pero con el indice posterior
        queue[ind]=value;
        ind+=1
        print('*' * 25)
        print("El cliente se ha formado. ")
        print('*' * 25)
        return queue, value, ind
    elif ind ==2:                   #se repite el mismo procedimiento, pero con el indice posterior
        queue[ind]=value;
        ind+=1
        print('*' * 25)
        print("El cliente se ha formado. ")
        print('*' * 25)
        return queue, value, ind
    elif ind ==3:                #se repite el mismo procedimiento, pero con el indice posterior
        queue[ind]=value;
        ind+=1
        print('-' * 25)
        print("El cliente se ha formado. ")
        print ('-' * 25)
```

```
        return queue, value, ind
    elif ind ==4:              #se repite el mismo procedimiento, pero con el indice posterior
        queue[ind]=value;
        ind+=1
        print('*' * 25)
        print("El cliente se ha formado. ")
        print('*' * 25)
        return queue, value, ind
    else:                      #una vez se llenan los 5 espacios se imprime una advertencia de que no se puede
formar mas gente
        print('*' * 25)
        print("La fila se ha llenado, espere a que alguien termine de pagar. ")
        print('*' * 25)
        return queue, value, ind


def pagar(queue, ind):         #creacion del metodo pagar/pop que recibe parametros de la cola y el indice
    queue[0]=queue[1]              #se iguala el indice de la colacon su posterior para ir sacando a las
personas que van pagando
    queue[1]=queue[2]              #y por lo tanto tener espacio para que otros se formen
    queue[2]=queue[3]
    queue[3]=queue[4]
    queue[4]= 'Empty'
    ind-=1                     #se decrese el indice para que sus valores vayan saliendo
    print('*' * 25)
    print(queue)                   #se imprime la fila
    print('*' * 25)
    return queue, ind              #regresa valores de la cola y el indice

def peek(queue,ind):               #creacion del metodo peek que recibe valores de la cola y el indice
    if queue[0]=='Empty':          #condicion que dice, mientras el elemento con indice 0 de la cola sea
igua a 'empty'
        print('*' * 25)            #se desplegara un mensaje advirtiendo que ya no hay gente formada
        print("No hay nadie formado. ")
        print('*' * 25)
        return queue, ind          #se devuelven valores de la cola y de; indice
    else:
        print('*' * 25)
        print(queue[0])            #se imprimen los datos de la persona en el primer indice el cual es el '0'
        print('*' * 25)
        return queue, ind          # se devuelven valores de la cola y el indice

def peekAll(queue, ind):           #creacion del metodo peekAll que recibe parametros de la cola y el
indice
    print('*' * 25)
    print(queue)                   #se encarga de imprimir toda la cola, para mostrar los datos de la gente
formada
    print('*' * 25)
    return queue, ind              #devuelve valores de la cola y el indice
```

```
def Tienda(queue, ind):                #se crea el metodo tienda el cual es el menu y recibe parametros de la cola y el indice
   while True:
      print("MENU:")
      print("1.- Meter un cliente en la fila. ")            #se da a elegir al usuario que es lo que quiere realizar
      print("2.- Mostrar primer cliente de la fila. ")
      print("3.- Cobrar al primer cliente. ")
      print("4.- Mostrar fila. ")
      print("5.- Salir del programa. ")
      print('*' * 25)
      k=int(input("¿Qué desea hacer? "))      #se pide al usuario ingrese su eleccion

      if k==1:                  #se crea la condicion, si la eleccion es igual a 1 se pide al usuario ingrese el nombre y numero del cliente
         value=str(input("""Ingrese "Nombre del cliente y su numero" para ingresarlo a la fila. """))
         queue, value, ind = formar(queue, value, ind);      # se mandan la cola, los datos ingresados y el indice al metodo a llamar, en este caso el metodo formar
         print(queue)                    # se imprime la cola
         print('*' * 25)
      else:
         if k==2:           #si la eleccion es 2
            queue, ind = peek(queue, ind); # se manda a llamar al metodo peek
         else:
            if k==3:              #si la eleccion es 3
               queue, ind = pagar(queue, ind); #se manda a llamar al metodo pagar
               if ind < 0:        #y se crea una condicion, si el indice es menor que cero se dice que la fila esta vacioa
                  print("La fila esta vacia. ")
                  print('*' * 25)
            else:
               if k==4:    #si la eleccion es 4
                  queue, ind = peekAll(queue, ind);  #se manda a llamar al metodo peekAll
               else:
                  if k==5:    #si la eleccion es 5
                     print("Fin del progama.")       #se finaliza el programa
                     sys.exit()


Tienda(queue, ind)
```

Nota: Este Examen fue escrito intencionalmente sin acentos. La calificacion del examen depende del desempeño del aplicante, el tiempo limite del examen se establece por el profesor al momento de entregar el mismo.