Trabajo Práctico Nº 11
Tema: PILAS (LIFO)

1. Realizar la traza de los siguientes segmentos de código.

a)
```
int X=3;
int Y=5;
int Z=2;
Pila P=new Pila();
P.meter(X);
P.meter(4);
X=P.sacar();
P.meter(Y);
P.meter(3);
P.meter(Z);
X=P.sacar();
P.meter(2);
P.meter(X);
while(!P.estaVacia())
   {
   X=P.sacar();
   System.out.println(X);
   }
```

b)
```
int Y=1;
Pila P=new Pila();
P.meter(5);
P.meter(7);
int X=P.sacar();
X+=Y;
P.meter(X);
P.meter(Y);
P.meter(2);
Y=P.sacar();
X=P.sacar();
while(!P.estaVacia())
   {
   Y=P.sacar();
   System.out.println(Y);
   }
System.out.println("X = "+ X);
System.out.println("Y = "+ Y);
```

c)
```
Pila P1=new Pila();
Pila P2=new Pila();
int X;
for(int i=1; i<=10; i++)
   P1.meter(i);
while(!P1.estaVacia())
   {
   X=P1.sacar();
   if(X%2 == 0)
      P2.meter(X);
   }
while(!P2.estaVacia())
   {
   X=P2.sacar();
   System.out.println(X);
   }
```

d)
```
int I=1;
int J;
Pila P1=new Pila();
Pila P2=new Pila();
while(I * I < 50)
   {
   J=I * I;
   P1.meter(J);
   I++;
   }
for(int i=1; i<=5; i++)
   {
   J=P1.sacar();
   P2.meter(J);
   }
I=P1.sacar();
int K;
for(int j=1; j<=I; j++)
   {
   K=P2.sacar();
   P1.meter(K);
   }
while(!P1.estaVacia())
   {
   I=P1.sacar();
   System.out.println(I);
   }
```

2. Dado el objeto pila P que contiene dos atributos: un arreglo de caracteres de dimensión 5 y CIMA un entero. C es una variable de tipo carácter. Para cada ejemplo de los que siguen, mostrar el resultado de la operación sobre la pila. Si ocurre desbordamiento o desbordamiento negativo, comprobar el caso correspondiente; si no mostrar el nuevo contenido del arreglo, CIMA y C (Nota: algunos valores del arreglo pueden no estar en la pila).

a)

P

‘A’	‘B’	‘C’	‘D’	‘E’
[0]	[1]	[2]	[3]	[4]

CIMA = 4 y C = ‘F’

P.meter(C); ¿Desbordamiento?..... ¿Desbordamiento Negativo? .....

P

[0]	[1]	[2]	[3]	[4]

CIMA = … y C = …

b)

P

‘S’	‘T’	‘U’	‘V’	‘W’
[0]	[1]	[2]	[3]	[4]

CIMA = 3 y C = ‘A’

P.meter(C); ¿Desbordamiento?..... ¿Desbordamiento Negativo? .....

P

[0]	[1]	[2]	[3]	[4]

CIMA = … y C = …

c)

P

‘B’	‘C’	‘D’	‘F’	‘Q’
[0]	[1]	[2]	[3]	[4]

CIMA = 0

C=P.sacar(); ¿Desbordamiento?..... ¿Desbordamiento Negativo? .....

P

[0]	[1]	[2]	[3]	[4]

CIMA = … y C = …

d)

P

‘X’	‘Y’	‘Z’	‘A’	‘B’
[0]	[1]	[2]	[3]	[4]

CIMA = 4 y C = ‘B’

P.meter(C); ¿Desbordamiento?..... ¿Desbordamiento Negativo? .....

P

[0]	[1]	[2]	[3]	[4]

CIMA = … y C = …

e)

P

‘B’	‘C’	‘D’	‘F’	‘Q’
[0]	[1]	[2]	[3]	[4]

CIMA = -1

C=P.sacar(); ¿Desbordamiento?..... ¿Desbordamiento Negativo? .....

P

[0]	[1]	[2]	[3]	[4]

CIMA = … y C = …

3. Realice un programa que dado el ingreso de una palabra como cadena de caracteres, permita visualizarla en forma inversa.

4. Escribir una versión del método meter() que cumpla con la siguiente especificación:

boolean meter(*nuevoelemento*)

Añade un nuevo elemento a la pila, si la pila no esta llena retorna *falso*, caso contrario retorna *verdadero*. Aplicar el mismo caso, pero esta vez, para el método sacar().

5. Utilizando la clase Pila implementada solo con arreglo, escribir un programa que permita meter y sacar elementos de la pila. Cada vez que se realice una operación deberá informar cuantos elementos hay en la pila y cuantos elementos le faltan para llegar al máximo.

6. Agregar a la clase Pila el método elementoCima() que retorne el elemento cima sin cambiar la pila. Nota: no olvidar que si bien la pila se implementa como un arreglo, no debería manejarse como tal.

7. Agregar a la clase Cadena el método esPalindromo() que retorna verdadero si la secuencia de caracteres se lee igual de izquierda a derecha y viceversa.
Ejemplo: ABLE WAS I ERE I SAW ELBA es palíndromo.

8. Realice un programa que lea una expresión aritmética y determine si tiene correctamente colocados los separadores (),{},[]. Obtener la expresión como un String.
Ejemplo:

[(a % b) * c]	→	correcto
2 * (a + b)]] / 2.5 + x	→	incorrecto
{(c - d) * [(c + a / t]	→	incorrecto

Nota: puede utilizar una pila para registrar los diferentes tipos de agrupación separadores. En cualquier momento que encuentre un signo de estos abriendo la expresión lo mete en la pila y cada vez que encuentre un signo terminal examina la pila. Si los signos coinciden, continúa testeando, caso contrario la expresión no será correcta.

9. Un estacionamiento de autos contiene una sola línea, la cual puede guardar hasta 10 autos. Existe solamente una entrada/salida al estacionamiento en uno de los extremos de la línea. Si un usuario llega a retirar su auto que no esta cerca de la salida, todos los autos que están bloqueando su salida son retirados, el auto del usuario sale, y los otros autos quedan en el orden que estaban originalmente. Escriba un programa que procese un grupo de entradas. Cada entrada contiene una 'e' para llegada o 's' para salida y una placa con el número de patente. Se asume que los autos llegan y salen en el orden especificado en la entrada. El programa debe mostrar un mensaje cada vez que sale o entra un auto. Cuando un auto llega, el mensaje debe especificar si hay lugar o no para el auto. Si no hay lugar, el auto saldrá sin entrar en el estacionamiento. Cuando un auto ha estado dentro del estacionamiento y sale, el mensaje debe incluir el número de veces que fue movido el auto para permitir que otros autos salieran.

10. Usar los métodos sacar(), meter(), estaVacia() para hacer las siguientes operaciones:
a) Asignar a X el segundo elemento desde la parte superior de la pila, dejando la pila sin sus dos elementos de la parte superior.
b) Asignar a X el segundo elemento desde la parte superior de la pila, sin modificarla.
c) Desde un entero positivo N, asignar a X el N-ésimo elemento desde la parte superior de la pila, dejando la pila sin sus N elementos de la parte superior.

d) Dado un entero positivo N, asignar a X el N-ésimo elemento desde la parte superior de pila, sin modificarla.
e) Asignar a X el elemento fondo de la pila, dejando la pila vacía.
f) Asignar a X el elemento fondo de la pila, sin modificarla.

11. Agregar a la clase Pila un método que retorne una copia exacta de una pila. Nota: no olvidar que si bien la pila se implementa como un arreglo, no debería manejarse como tal.

12. Se desea implementar la clase DoblePila de enteros. Una doble pila, es una pila en la que se pueden insertar y extraer elementos por ambos extremos. Esta doble pila contendrá como máximo 200 elementos. Los elementos a almacenar están comprendidos entre 500 y 1000. Suponiendo que valor es el elemento a guardar o extraer sabemos que:

cuando 500 <= valor <= 750 los elementos se ingresan y extraen por un extremo.
cuando 750 < valor <= 1000 los elementos se ingresan y extraen por el otro extremo.

En una determinada posición de la doble pila nunca habrá más de un elemento, es decir, la cantidad total de elementos de la doble pila no excederá al máximo.
a) Realice el esquema gráfico de la doble pila.
b) Codificar la clase DoblePila que incluya la declaración de atributos y las operaciones básicas para esta clase (constructor, meter(), sacar(), estaLlena(), estaVacia()).
c) Escribir un programa, que pruebe la clase DoblePila.

13. Un mapa de rutas se puede representar mediante una matriz simétrica MM de N x N elementos enteros, donde los valores 1 a N representan los pueblos/ciudades que aparecen en el mapa.
Los elementos de la matriz son tales que MM(i, j) = 0 si no hay conexión directa entre el pueblo i y el pueblo j. MM(i, j) = d si hay conexión directa entre el pueblo i y el pueblo j, y su distancia es d.
Con esta representación del mapa queremos escribir un programa que simule el mapa descrito y que tenga como entrada dos pueblos (origen, destino) entre los que no hay conexión directa; decida si hay un camino que pase por los pueblos del mapa y determine la distancia de ese camino. Utilizar una pila para ir almacenando los pueblos que van formando el camino recorrido y poder volver atrás si se alcanza un pueblo desde el que no se puede proseguir la ruta y probar con otra ruta.

14. Escribir un método para determinar si una secuencia de caracteres de entrada es de la forma: X & Y. Donde X es una cadena de caracteres, e Y es la cadena inversa, siendo & el carácter separador.

15. Escribir un método para determinar si una secuencia de caracteres de entrada es de la forma: A # B # C # ... Donde cada una de las cadenas A, B, C, ... son de la forma X & Y, que a su vez estarán separadas por el carácter #.

16. Una Pila de Object permite utilizar la misma clase para trabajar con objetos enteros, reales, cadenas, etc. Pruebe la siguiente clase con objetos de diferente clase (Integer, Float, Char, etc.)