EjerciciosListassimples

1. Defina un menú de opciones que permita gestionar la antigüedad de los empleados de una empresa:
   1. Insertar un nodo al inicio
   2. Insertar un nodo alfinal
   3. Mostrar el contenido de la lista
   4. Eliminar un nodo del inicio
   5. Eliminar un nodo del final

La información que guarda cada nodo es un valor entero que representa la antigüedad del empleado.

//CLASE NODO

public class Nodo {

int antiguedad;

Nodo siguiente;

public Nodo(int antiguedad) {

this.antiguedad = antiguedad;

this.siguiente = null;

}

}

//CLASE LISTA DE ANTIGUEDADES

public class ListaAntiguedades {

private Nodo inicio;

private Nodo fin;

public ListaAntiguedades() {

inicio = null;

fin = null;

}

public void insertarAlInicio(int antiguedad) {

Nodo nuevoNodo = new Nodo(antiguedad);

if (inicio == null) {

inicio = nuevoNodo;

fin = nuevoNodo;

} else {

nuevoNodo.siguiente = inicio;

inicio = nuevoNodo;

}

}

public void insertarAlFinal(int antiguedad) {

Nodo nuevoNodo = new Nodo(antiguedad);

if (inicio == null) {

inicio = nuevoNodo;

fin = nuevoNodo;

} else {

fin.siguiente = nuevoNodo;

fin = nuevoNodo;

}

}

public void eliminarNodoInicio() {

if (inicio != null) {

inicio = inicio.siguiente;

}

}

public void eliminarNodoFinal() {

if (inicio != null) {

if (inicio == fin) {

inicio = null;

fin = null;

} else {

Nodo actual = inicio;

while (actual.siguiente != fin) {

actual = actual.siguiente;

}

actual.siguiente = null;

fin = actual;

}

}

}

public void mostrarContenido() {

Nodo actual = inicio;

while (actual != null) {

System.out.print(actual.antiguedad + " -> ");

actual = actual.siguiente;

}

System.out.println("null");

}

}

//CLASE MAIN GESTION DE ANTIGUEDADES

import java.util.Scanner;

public class GestionDeAntiguedadesEj1 {

public static void main(String[] args) {

// TODO code application logic here

Scanner scanner = new Scanner(System.in);

ListaAntiguedades lista = new ListaAntiguedades();

while (true) {

System.out.println("\nMenú de opciones:");

System.out.println("a. Insertar un nodo al inicio");

System.out.println("b. Insertar un nodo al final");

System.out.println("c. Mostrar el contenido de la lista");

System.out.println("d. Eliminar un nodo del inicio");

System.out.println("e. Eliminar un nodo del final");

System.out.println("x. Salir");

System.out.print("Elija una opción: ");

String opcion = scanner.next();

if (opcion.equals("a")) {

System.out.print("Ingrese la antiguedad del empleado: ");

int antiguedad = scanner.nextInt();

lista.insertarAlInicio(antiguedad);

} else if (opcion.equals("b")) {

System.out.print("Ingrese la antiguedad del empleado: ");

int antiguedad = scanner.nextInt();

lista.insertarAlFinal(antiguedad);

} else if (opcion.equals("c")) {

System.out.println("Contenido de la lista:");

lista.mostrarContenido();

} else if (opcion.equals("d")) {

lista.eliminarNodoInicio();

System.out.println("Nodo del inicio eliminado.");

} else if (opcion.equals("e")) {

lista.eliminarNodoFinal();

System.out.println("Nodo del final eliminado.");

} else if (opcion.equals("x")) {

break;

} else {

System.out.println("Opción no válida. Intente de nuevo.");

}

}

scanner.close();

}

}

1. A la aplicación anterior agregar las siguientes opciones.
   1. Mostrar el contenido de la lista ordenada de menor a mayor.
   2. Mostrar en pantalla la cantidadd eempleados con una antigüedad superior a 10 años.
   3. Calcular la antigüedad promedio de los empleados.
   4. Mostrar cuanto cobra el empleado con másantigüedad.Para ello tener en cuenta lo siguiente:
      1. El salario base es de $250000
      2. Si la antigüedad es superior a10 años cobra un adicional de $5000 por cada año excedido.
      3. Si la antigüedad es superior a 20 años cobra un adicional de

$10000por cada año excedido

Ej.:un empleado con 15 años de antigüedad cobra

$25000+(5\*$5000)=$275000

//CLASE NODO

public class Nodo {

int antiguedad;

Nodo siguiente;

public Nodo(int antiguedad) {

this.antiguedad = antiguedad;

this.siguiente = null;

}

//CLASE LISTA DE ANTIGUEDADES

import java.util.ArrayList;

import java.util.Collections;

public class ListaDeAntiguedades {

private Nodo inicio;

private Nodo fin;

public ListaDeAntiguedades(){

inicio = null;

fin = null;

}

public void insertarAlInicio(int antiguedad) {

Nodo nuevoNodo = new Nodo(antiguedad);

if (inicio == null) {

inicio = nuevoNodo;

fin = nuevoNodo;

} else {

nuevoNodo.siguiente = inicio;

inicio = nuevoNodo;

}

}

public void insertarAlFinal(int antiguedad) {

Nodo nuevoNodo = new Nodo(antiguedad);

if (inicio == null) {

inicio = nuevoNodo;

fin = nuevoNodo;

} else {

fin.siguiente = nuevoNodo;

fin = nuevoNodo;

}

}

public void eliminarNodoInicio() {

if (inicio != null) {

inicio = inicio.siguiente;

}

}

public void eliminarNodoFinal() {

if (inicio != null) {

if (inicio == fin) {

inicio = null;

fin = null;

} else {

Nodo actual = inicio;

while (actual.siguiente != fin) {

actual = actual.siguiente;

}

actual.siguiente = null;

fin = actual;

}

}

}

public void mostrarContenido() {

Nodo actual = inicio;

while (actual != null) {

System.out.print(actual.antiguedad + " -> ");

actual = actual.siguiente;

}

System.out.println("null");

}

public void mostrarOrdenado() {

ArrayList<Integer> antiguedades = obtenerAntiguedades();

Collections.sort(antiguedades); // Ordena la lista de antigüedades

System.out.println("Lista ordenada de menor a mayor:");

for (int antiguedad : antiguedades) {

System.out.print(antiguedad + " -> ");

}

System.out.println("null");

}

public ArrayList<Integer> obtenerAntiguedades() {

ArrayList<Integer> antiguedades = new ArrayList<>();

Nodo actual = inicio;

while (actual != null) {

antiguedades.add(actual.antiguedad);

actual = actual.siguiente;

}

return antiguedades;

}

public int calcularAntiguedadPromedio() {

int totalAntiguedad = 0;

Nodo actual = inicio;

while (actual != null) {

totalAntiguedad += actual.antiguedad;

actual = actual.siguiente;

}

int cantidadEmpleados = contarNodos();

if (cantidadEmpleados > 0) {

return totalAntiguedad / cantidadEmpleados;

} else {

return 0;

}

}

public int calcularSalarioEmpleadoMaxAntiguedad() {

Nodo actual = inicio;

int maxAntiguedad = 0;

while (actual != null) {

if (actual.antiguedad > maxAntiguedad) {

maxAntiguedad = actual.antiguedad;

}

actual = actual.siguiente;

}

int salarioBase = 250000;

int salarioAdicional = 0;

if (maxAntiguedad > 10) {

salarioAdicional = (maxAntiguedad - 10) \* 5000;

}

if (maxAntiguedad > 20) {

salarioAdicional += (maxAntiguedad - 20) \* 10000;

}

return salarioBase + salarioAdicional;

}

public int contarNodos() {

Nodo actual = inicio;

int contador = 0;

while (actual != null) {

contador++;

actual = actual.siguiente;

}

return contador;

}

}

//CLASE MAIN GESTION DE ANTIGUEDADES

import java.util.ArrayList;

import java.util.Scanner;

public class GestionDeAntiguedadesEj2 {

public static void main(String[] args) {

// TODO code application logic here

Scanner scanner = new Scanner(System.in);

ListaDeAntiguedades lista = new ListaDeAntiguedades();

while (true) {

System.out.println("\nMenú de opciones:");

System.out.println("a. Insertar un nodo al inicio");

System.out.println("b. Insertar un nodo al final");

System.out.println("c. Mostrar el contenido de la lista");

System.out.println("d. Eliminar un nodo del inicio");

System.out.println("e. Eliminar un nodo del final");

System.out.println("f. Mostrar la lista ordenada de menor a mayor");

System.out.println("g. Mostrar la cantidad de empleados con antigüedad superior a 10 años");

System.out.println("h. Calcular la antigüedad promedio de los empleados");

System.out.println("i. Calcular el salario del empleado con más antiguedad");

System.out.println("x. Salir");

System.out.print("Elija una opción: ");

String opcion = scanner.next();

if (opcion.equals("a")) {

System.out.print("Ingrese la antiguedad del empleado: ");

int antiguedad = scanner.nextInt();

lista.insertarAlInicio(antiguedad);

} else if (opcion.equals("b")) {

System.out.print("Ingrese la antiguedad del empleado: ");

int antiguedad = scanner.nextInt();

lista.insertarAlFinal(antiguedad);

} else if (opcion.equals("c")) {

System.out.println("Contenido de la lista:");

lista.mostrarContenido();

} else if (opcion.equals("d")) {

lista.eliminarNodoInicio();

System.out.println("Nodo del inicio eliminado.");

} else if (opcion.equals("e")) {

lista.eliminarNodoFinal();

System.out.println("Nodo del final eliminado.");

} else if (opcion.equals("f")) {

lista.mostrarOrdenado();

} else if (opcion.equals("g")) {

int cantidadEmpleadosSuperiorA10Anios = contarAntiguedadesSuperioresA(lista, 10);

System.out.println("Cantidad de empleados con antigüedad superior a 10 años: " + cantidadEmpleadosSuperiorA10Anios);

} else if (opcion.equals("h")) {

int antiguedadPromedio = lista.calcularAntiguedadPromedio();

System.out.println("Antigüedad promedio de los empleados: " + antiguedadPromedio + " años");

} else if (opcion.equals("i")) {

int salarioEmpleadoMaxAntiguedad = lista.calcularSalarioEmpleadoMaxAntiguedad();

System.out.println("Salario del empleado con más antigüedad: $" + salarioEmpleadoMaxAntiguedad);

} else if (opcion.equals("x")) {

break;

} else {

System.out.println("Opción no válida. Intente de nuevo.");

}

}

scanner.close();

}

public static int contarAntiguedadesSuperioresA(ListaDeAntiguedades lista, int valor) {

ArrayList<Integer> antiguedades = lista.obtenerAntiguedades();

int contador = 0;

for (int antiguedad : antiguedades) {

if (antiguedad > valor) {

contador++;

}

}

return contador;

}

}

1