

Prueba Sumativa 1 - Parte 1

Programación Orientada a Objetos

Nombre		RUT	
Paralelo	C1 A. Paolini () C2 C. Rabi ()		

Antecedentes generales:

Puntaje total de la prueba/Puntos para nota aprobatoria (4.0)	1 puntos 0.6 puntos	Duración de la prueba	3 horas (1 hora + 10 min break + 2 horas)
Resultados de Aprendizaje a evaluar	1. Aplicar técnicas de ingeniería de software en la creación de software legible, mantenible y testeable. 2. Aplicar técnicas de programación orientada al objeto en la resolución de problemas. 3. Crear tipos de datos abstractos con bajo acoplamiento entre la implementación y su comportamiento que permitan la resolución problemas. 4. Analizar las relaciones causa efecto de los procesos en estudio. 6. Seleccionar los procesos, técnicas y herramientas adecuados de acuerdo a los requerimientos.		
Fecha de entrega de resultados	17 octubre 2025		

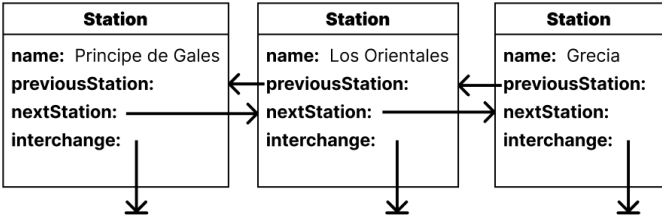
Instrucciones:

- Esta evaluación tiene 2 partes y 5 páginas (4 en primera parte y 1 en la segunda). Compruebe que dispone de todas las páginas.
- Lea la prueba completamente DOS veces antes de hacer cualquier pregunta
- Durante la prueba no se puede utilizar: teléfonos móviles y apuntes. Está prohibido intentar conectarse a internet de cualquier manera (excepto a Campus Virtual, y solo para subir su solución). Si es sorprendido obtendrá la calificación mínima. Tampoco puede utilizar dispositivos de almacenamiento externos o cualquier otro dispositivo como relojes inteligentes, tables, audífonos, etc.
- Una prueba respondida correctamente en un 60%, de acuerdo con las ponderaciones asignadas, corresponde a una nota 4,0.
- La prueba es individual, cualquier sospecha de copia será calificada con la nota mínima y el caso será remitido al comité de ética.
- En su espacio personal no debe haber nada más que hojas de papel en blanco, lápiz y goma. El resto de sus implementos debe guardarlos dentro de su mochila/bolso y ésta debe posicionarse al frente debajo de la pizarra. Si leyó hasta este punto, dibuje 3 rombos consecutivos al final de esta página.
- Los estudiantes quienes se les compruebe falta de honestidad académica o cualquier otro acto contrario a las normas de permanencia universitaria o al espíritu universitario, serán sancionados, según sea la gravedad de la falta, con medidas desde la amonestación verbal hasta la suspensión o pérdida de la condición de estudiante. Los estudiantes expulsados no podrán volver a ingresar a ninguna carrera, programa o curso de la institución. El estudiante que incurriere en falta de honestidad, durante la realización de un proceso evaluativo, será calificado con la nota mínima 1,0.

Acepto las condiciones firmando: _____

Problema 1. Ruteo (0.33 puntos)

Rutee el siguiente código e indique la salida esperada. No es necesario que dibuje los objetos, pero le será de gran ayuda para orientarse mejor. Puede utilizar este ejemplo de guía.



```
POO - Untitled-2

1 public class App {
2     public static void main(String[] args) {
3         Line one = new Line(1);
4         Line two = new Line(2);
5         Line six = new Line(6);
6         Line four = new Line(4);
7         try {
8             one.addStations("San Pablo,Los Heroes,Baquedano,Los Leones,El Golf,Los Dominicos");
9             two.addStations("Vespucio Norte,Los Heroes,Franklin,Hospital El Pino");
10            six.addStations("Cerrillos,Franklin,Bío Bío,Ñuble,Ñuñoa,Los Leones");
11            one.printStationsForward();
12            two.printStationsBackward();
13            one.setInterchangeStations(two, "Los Heroes");
14            two.setInterchangeStations(six, "Franklin");
15            six.setInterchangeStations(one, "Los Leones");
16            one.roam(8);
17            six.roam(5);
18            four.addStations("Tobalaba,Plaza de Puente Alto");
19        } catch (Exception e) {
20            System.out.println(e.getMessage());
21        }
22    }
23 }

24
25 public class Station {
26     private String name;
27     private Station previousStation;
28     private Station nextStation;
29     private Station interchange;
30     public Station(String name) {
31         this.name = name;
32     }
33     public Station getPreviousStation() {
34         return previousStation;
35     }
36     public void setPreviousStation(Station previousStation) {
37         this.previousStation = previousStation;
38     }
39     public Station getNextStation() {
40         return nextStation;
41     }
42     public void setNextStation(Station nextStation) {
43         this.nextStation = nextStation;
44     }
45     public Station getInterchange() {
46         return interchange;
47     }
48     public void setInterchange(Station interchange) {
49         this.interchange = interchange;
50     }
51     public String getName() {
52         return name;
53     }
54 }
```

POO - Untitled-2

```
56 public class Line {
57     private int number;
58     private Station terminalStationBegin;
59     private Station terminalStationEnd;
60     public Line(int number) {
61         this.number = number;
62     }
63     public void addStations(String names) {
64         String[] stationNames = names.split(",");
65         int stationCount = stationNames.length;
66         if (stationCount < 3) {
67             throw new IllegalArgumentException("Faltan estaciones");
68         }
69         this.terminalStationBegin = new Station(stationNames[0]);
70         Station aux = this.terminalStationBegin;
71         for (int i = 1; i < stationNames.length - 1; i++) {
72             aux.setNextStation(new Station(stationNames[i]));
73             aux.getNextStation().setPreviousStation(aux);
74             aux = aux.getNextStation();
75         }
76         this.terminalStationEnd = new Station(stationNames[stationCount - 1]);
77         aux.setNextStation(terminalStationEnd);
78         this.terminalStationEnd.setPreviousStation(aux);
79     }
80     public void printStationsForward() {
81         System.out.println("Linea " + this.number + " - Dirección " + terminalStationEnd.getName());
82         Station aux = terminalStationBegin;
83         while (aux.getNextStation() != null) {
84             System.out.print(aux.getName() + " > ");
85             aux = aux.getNextStation();
86         }
87         System.out.print(aux.getName());
88         System.out.println();
89     }
90     public void printStationsBackward() {
91         System.out.println("Linea " + this.number + " - Dirección " + terminalStationBegin.getName());
92         Station aux = terminalStationEnd;
93         while (aux.getPreviousStation() != null) {
94             System.out.print(aux.getName() + " > ");
95             aux = aux.getPreviousStation();
96         }
97         System.out.print(aux.getName());
98         System.out.println();
99     }
}
```

```
POO - Untitled-2

100 public Station findStation(String station) {
101     Station aux = terminalStationBegin;
102     while (aux.getNextStation() != null) {
103         if (aux.getName().equals(station)) {
104             return aux;
105         }
106         aux = aux.getNextStation();
107         if (aux.getName().equals(station)) {
108             return aux;
109         }
110     }
111     return null;
112 }
113 public void setInterchangeStations(Line line, String name) {
114     Station stationA = this.findStation(name);
115     Station stationB = line.findStation(name);
116     stationA.setInterchange(stationB);
117     stationB.setInterchange(stationA);
118 }
119 public void roam(int stops) {
120     System.out.println("Deambulando desde " + terminalStationBegin.getName());
121     Station aux = terminalStationBegin;
122     int stationCounter = 0;
123     while (aux.getNextStation() != null && stationCounter < stops - 1) {
124         System.out.print(aux.getName() + " - ");
125         if (aux.getInterchange() == null) {
126             aux = aux.getNextStation();
127         } else {
128             aux = aux.getInterchange().getNextStation();
129         }
130         stationCounter++;
131     }
132     System.out.print(aux.getName());
133     System.out.println();
134 }
135 }
```