2019-EDA ☑ Exámenes

Exámenes

Tema 1 - S3: Cuestiones sobre el Modelo Lista Con PI y su uso

Volver a la Lista de Exámenes

Parte 1 de 2 - Introducción 0.0/ 0.0 Puntos

En esta actividad tendrás que trabajar con las siguientes clases del proyecto BlueJ *eda*:

- Programas *TestListaDeLaCompra y TestListaDeLaCompraConPI* del paquete *tema1*
- Clases *LEGLista* del paquete *lineales* y *Lista, ListaConPI, Pila y Cola* de*l paquete modelos*

No hay preguntas

Parte 2 de 2 - Cuestiones 13.58/ 20.0 Puntos

Preguntas 1 de 8

1.5/ 1.5 Puntos

LEGLista<E> es la implementación de la interfaz *Lista<E>* disponible hasta la fecha. Tras analizar estas clases, responde a la siguiente cuestión.

El programa *TestListaDeLaCompra* usa una *Lista<String>* , implementada mediante una *LEGLista<String>*, para representar y gestionar la confección de una lista de la compra; por tanto, ...

- ✔ ○ **El coste de insertar un producto en el final de la lista de la compra es lineal con la talla de la Lista y el de buscar *perejil* es cuadrático con la talla de la lista.**
- ○ **Tanto el coste de insertar un producto al final de la lista de la compra como el de buscar *perejil* son lineales con la talla de la lista.**
- ○ **El coste de insertar un producto al final de la lista de la compra es constante y el de buscar *perejil* es lineal con la talla de la lista.**

Respuesta correcta:A

Preguntas 2 de 8

0.0/ 1.5 Puntos

La interfaz *ListaConPI<E>* representa una Lista en la que, en cada instante, solo es posible acceder al elemento que ocupa su Punto de Interés (PI). Suponiendo que la clase *LEGListaConPI <E>* es una Implementación eficiente de esta interfaz, selecciona aquellas opciones (una o más) que responden correctamente a la siguiente cuestión.

El programa *TestListaConPIDeLaCompra* usa una *ListaConPI<String>* , implementada mediante una *LEGListaConPI<String>*, para representar y gestionar la confección de una lista de la compra; por tanto, ...

- [ ] Solo usando una Lista Con PI, el coste de insertar un producto en el final de la lista de la compra es constante y el de buscar *perejil* es lineal con la talla de la lista.
- [ ] ✖ Independientemente del Modelo Java de Lista que se use, tanto el coste de insertar un producto al final de la lista de la compra como el de buscar *perejil* son lineales con la talla de la lista.
- [ ] Independientemente del Modelo Java de Lista que usen, los programas *TestListaConPIDeLaCompra* y *TestListaDeLaCompra* tienen el mismo coste.
- [ ] Independientemente del Modelo Java de Lista que se use, el coste de insertar un producto en el final de la lista de la compra es lineal con la talla de la Lista y el de buscar *perejil* es cuadrático con la talla de la lista.
- [ ] ✔ Gracias a la inclusión del PI en el Modelo de una Lista, el programa *TestListaConPIDeLaCompra* es mucho más eficiente que el de *TestListaDeLaCompra*.

Respuesta correcta:A, E

Preguntas 3 de 8

3.0/ 3.0 Puntos

Se quiere modificar el código actual del programa *TestListaConPIDeLaCompra* para, si no está, apuntar *perejil* al principio de la lista (en vez de al final). Para ello basta añadir una instrucción a una de sus líneas...

- **Indica el número de línea donde hay que añadir dicha instrucción:** ✔ **24**
- **Indica, siguiendo las convenciones de código Java, la instrucción a añadir:** ✔ **l.inicio();**
- **Indica si añadirías dicha instrucción ANTES o DESPUÉS de la instrucción que ya hay en dicha línea:** ✔ **ANTES**

NOTA QUE puedes comprobar la validez de tus respuestas modificando el código del programa y ejecutándolo.

Respuesta correcta:24, l.inicio();, ANTES|antes

Preguntas 4 de 8

1.2/ 3.0 Puntos

La línea número 12 del código actual del programa *TestListaConPIDeLaCompra* está vacía. Escribe en ella la siguiente instrucción: *l.insertar("perejil").* Obviamente, con ello *perejil* queda apuntado al principio de la lista de la compra.

Supón ahora que cambias de idea y decides borrar *perejil* del principio de la lista y apuntarlo ANTES que *cerezas*. Para ello basta con: (a) añadir una instrucción en una línea del programa; (b) modificar la guarda del actual bucle de búsqueda de *perejil* (línea 22); (c) substituir el *actual if* que empieza en la línea 23 por la instrucción *l.insertar(e)* , pues al tener la completa seguridad de que *cerezas* es un elemento de la lista *l,* la búsqueda descrita en la línea 22 pasa a tener garantía de éxito.

- **Indica el número de línea donde hay que añadir la nueva instrucción a las ya existentes: ✖14**
- **Indica, siguiendo las convenciones de código Java, la nueva instrucción a añadir: ✖l.insertar(perejil);**
- **Indica si añadirías dicha instrucción ANTES o DESPUÉS de las que ya hay en dicha línea: ✔DESPUÉS**
- **Indica la única modificación que es imprescindible hacer en la guarda del actual bucle de búsqueda de *perejil* (línea 22): cambiar la variable ✔e por el literal de *String* " ✖perejil "**

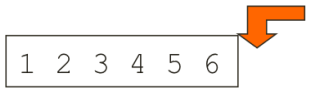
NOTA QUE puedes comprobar la validez de tus respuestas modificando el código del programa y ejecutándolo.

Respuesta correcta:21, l.eliminar();, DESPUÉS|después|DESPUES|despues, e, cerezas

Preguntas 5 de 8

2.25/ 3.0 Puntos

Sea la *ListaConPI* de *Integer l* que aparece en la imagen:



Responde a las siguientes preguntas completando los huecos que aparecen en ellas.

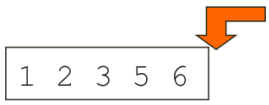
- **SÍ o NO, ¿se produce algún error al ejecutar la instrucción *l.eliminar()*?: ✔SÍ .**
- **¿Cuál es el resultado de ejecutar la instrucción *l.esFin()*?: ✖True .**
- **SÍ o NO, ¿se modifica el estado de la lista al ejecutar la instrucción *l.fin()*?: ✔NO .**
- **Al ejecutar la instrucción *l.inicio()* el elemento de la Lista sobre el que se sitúa el PI es el ✔1 .**

Respuesta correcta:SÍ|SI|si, true, NO|no, 1

Preguntas 6 de 8

3.0/ 3.0 Puntos

Sea la *ListaConPI* de *Integer* *l* que aparece en la imagen:



Completa los huecos que figuran en las siguientes frases.

- **Tras ejecutar la instrucción *l.inicio()*, ¿qué instrucción se debe aplicar 3 veces consecutivas para situar el PI de *l* sobre el 5? ✔l.siguiente(); .**
- **Situado su PI sobre el 5, TRAS ejecutar la instrucción *l.insertar(new Integer(4));* los elementos de la lista *l* son, en este orden: ✔1 , ✔2 , ✔3 , ✔4 , ✔5 y ✔6 . Además, el PI de *l* está situado sobre el ✔5 .**

Si ahora se ejecuta *l.eliminar();* los elementos de *l* serán ✔1 , ✔2 , ✔3 , ✔4 y ✔6 y su PI estará situado sobre el ✔6 .

- **Si el PI de *l* está situado sobre el 5, TRAS ejecutar las instrucciones *l.eliminar(); l.insertar(new Integer(4)* ¿el estado de la lista *l* sería el mismo (elementos y PI) que tras ejecutar *l.insertar(new Integer(4) ; l.eliminar()*? Responde SÍ o NO: ✔SÍ .**

Respuesta correcta:l.siguiente()|l.siguiente();, 1, 2, 3, 4, 5, 6, 5, 1, 2, 3, 4, 6, 6, SÍ|SI|sí|si|Sí|Si

Preguntas 7 de 8

1.32/ 2.5 Puntos

Recordando que el modelo de una Pila es LIFO, abre las interfaces *ListaConPI* y *Pila* del proyecto BlueJ *eda* y transforma la primera en la segunda completando los huecos de las siguientes cuestiones:

- **SÍ o NO, ¿eliminarías el método *insertar* de *ListaConPI* en la interfaz *Pila*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✖-**
- **SÍ o NO, ¿eliminarías el método *eliminar* de *ListaConPI* en la interfaz *Pila*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✖-**
- **SÍ o NO, ¿eliminarías el método *inicio* de *ListaConPI* en la interfaz *Pila*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✔-**
- **SÍ o NO, ¿eliminarías el método *siguiente* de *ListaConPI* en la interfaz *Pila*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✔-**
- **SÍ o NO, ¿eliminarías el método *fin* de *ListaConPI* en la interfaz *Pila*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✔-**
- **SÍ o NO, ¿eliminarías el método *recuperar* de *ListaConPI* en la interfaz *Pila*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✖-**
- **SÍ o NO, ¿eliminarías el método *esFin* de *ListaConPI* en la interfaz *Pila*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✔-**
- **SÍ o NO, ¿eliminarías el método *esVacia* de *ListaConPI* en la interfaz *Pila*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✖-**
- **SÍ o NO, ¿eliminarías el método *talla* de *ListaConPI* en la interfaz *Pila*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Pila*: ✔-**
- **En su caso, la PRECONDICIÓN de los métodos que contiene ahora la interfaz *Pila* no puede ser la que tenían en *ListaConPI* sino *SII* ✖Lista**

Respuesta correcta:NO|No|no, void apilar(E e)|void apilar(E e);, NO|No|no, E desapilar()|E desapilar();, SÍ|SI|sí|si|Sí|Si, -, SÍ|SI|sí|si|Sí|Si, -, SÍ|SI|sí|si|Sí|Si, -, NO|No|no, E tope()|E tope();, SÍ|SI|sí|si|Sí|Si, -, NO|No|no, boolean esVacia()|boolean esVacia();, SÍ|SI|sí|si|Sí|Si, -, !esVacia()

Preguntas 8 de 8

1.32/ 2.5 Puntos

Recordando que el modelo de una Cola es FIFO, abre las interfaces *ListaConPI* y *Cola* del proyecto BlueJ *eda* y transforma la primera en la segunda completando los huecos de las siguientes cuestiones:

- **SÍ o NO, ¿eliminarías el método *insertar* de *ListaConPI* en la interfaz *Cola*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✖-**
- **SÍ o NO, ¿eliminarías el método *eliminar* de *ListaConPI* en la interfaz *Cola*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✖-**
- **SÍ o NO, ¿eliminarías el método *inicio* de *ListaConPI* en la interfaz *Cola*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✔-**
- **SÍ o NO, ¿eliminarías el método *siguiente* de *ListaConPI* en la interfaz *Cola*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✔-**
- **SÍ o NO, ¿eliminarías el método *fin* de *ListaConPI* en la interfaz *Cola*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✔-**
- **SÍ o NO, ¿eliminarías el método *recuperar* de *ListaConPI* en la interfaz *Cola*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✖-**
- **SÍ o NO, ¿eliminarías el método *esFin* de *ListaConPI* en la interfaz *Cola*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✔-**
- **SÍ o NO, ¿eliminarías el método *esVacia* de *ListaConPI* en la interfaz *Cola*?: ✖SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✖-**
- **SÍ o NO, ¿eliminarías el método *talla* de *ListaConPI* en la interfaz *Cola*?: ✔SÍ . En el siguiente hueco, si tu respuesta es SÍ escribe un guión y si es NO la cabecera que el método tendría en *Cola*: ✔-**
- **En su caso, la PRECONDICIÓN de los métodos que contiene ahora la interfaz *Cola* no puede ser la que tenían en *ListaConPI* sino *SII* ✖cOLA**

Respuesta correcta:NO|No|no, void encolar(E e)|void encolar(E e);, NO|No|no, E desencolar()|E desencolar();, SÍ|SI|sí|si|Sí|Si, -, SÍ|SI|sí|si|Sí|Si, -, SÍ|SI|sí|si|Sí|Si, -, NO|No|no, E primero()|E primero();, SÍ|SI|sí|si|Sí|Si, -, NO|No|no, boolean esVacia()|boolean esVacia();, SÍ|SI|sí|si|Sí|Si, -, !esVacia()