**Documentación**

**Lote de Pruebas**

Programación Avanzada 2013

* *Dado que el problema admite soluciones múltiples para un mismo input, las salidas esperadas en este documento son sólo casos posibles escritos para utilizar con el programa probador.*

|  |  |
| --- | --- |
|  | |
| Nombre | 00\_CONSIGNA.in |
| Tipo | Derivado de análisis. |
| Descripción | Caso derivado de la consigna del ejercicio dado. |
| Archivo de Entrada | 00\_CONSIGNA.in |
| Entrada | \*\*\*\*R\* |
| Archivo de Salida | 00\_CONSIGNA.out |
| Salida (esperada) | VRVARA |

|  |  |
| --- | --- |
|  | |
| Nombre | 01\_TRES\_BALDOSAS.in |
| Tipo | Derivado de análisis. |
| Descripción | Caso que verifica el funcionamiento del algoritmo con la menor cantidad de baldosas posibles. |
| Archivo de Entrada | 01\_TRES\_BALDOSAS.in |
| Entrada | \*V\* |
| Archivo de Salida | 01\_TRES\_BALDOSAS..out |
| Salida (esperada) | AVR |

|  |  |
| --- | --- |
|  | |
| Nombre | 02\_6\_BALDOSAS\_PRIMER\_BALDOSA\_PINTADA.in |
| Tipo | Derivado de análisis. |
| Descripción | Se verifica el funcionamiento dada la primer baldosa pintada. Cantidad total de baldosas: 6. |
| Archivo de Entrada | 02\_6\_BALDOSAS\_PRIMER\_BALDOSA\_PINTADA.in |
| Entrada | R\*\*\*\*\* |
| Archivo de Salida | 02\_6\_BALDOSAS\_PRIMER\_BALDOSA\_PINTADA.out |
| Salida (esperada) | RAVRAV |

|  |  |
| --- | --- |
|  | |
| Nombre | 03\_21\_BALDOSAS\_ULTIMA\_BALDOSA\_PINTADA.in |
| Tipo | Derivado de análisis. |
| Descripción | Se verifica el funcionamiento dada la última baldosa pintada. Cantidad total de baldosas: 21. |
| Archivo de Entrada | 03\_21\_BALDOSAS\_ULTIMA\_BALDOSA\_PINTADA.in |
| Entrada | \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*A |
| Archivo de Salida | 03\_21\_BALDOSAS\_ULTIMA\_BALDOSA\_PINTADA.out |
| Salida (esperada) | VRAVRAVRAVRAVRAVRAVRA |

|  |  |
| --- | --- |
|  | |
| Nombre | 04\_FATIGA\_255\_BALDOSAS\_PINTADA\_LA\_127.in |
| Tipo | Fatiga. |
| Descripción | Se intenta fatigar el algoritmo dando la cantidad mayor de baldosas admitidas y ubicando la baldosa pintada en la posición 127. |
| Archivo de Entrada | 04\_FATIGA\_255\_BALDOSAS\_PINTADA\_LA\_127.in |
| Entrada | Omitido por extensión. |
| Archivo de Salida | 04\_FATIGA\_255\_BALDOSAS\_PINTADA\_LA\_127.out |
| Salida (esperada) | Omitido por extensión. |

|  |  |
| --- | --- |
|  | |
| Nombre | 05\_FATIGA\_255\_BALDOSAS\_PINTADA\_LA\_ULTIMA.in |
| Tipo | Fatiga. |
| Descripción | Se intenta fatigar el algoritmo dando la cantidad mayor de baldosas admitidas y ubicando la baldosa pintada en la última posición. |
| Archivo de Entrada | 04\_FATIGA\_255\_BALDOSAS\_PINTADA\_LA\_ULTIMA.in |
| Entrada | Omitido por extensión. |
| Archivo de Salida | 04\_FATIGA\_255\_BALDOSAS\_PINTADA\_LA\_ULTIMA.out |
| Salida (esperada) | Omitido por extensión. |

2.B) Otros tipos de prueba son:

- Prueba Alfa: La realiza el fabricante sobre versiones no finales del software. Son las primeras pruebas realizadas sobre el programa a manos de los mismos programadores o personal de testing.

- Prueba Beta: La realiza el usuario final sobre versiones no finales del software. Estas pueden ser liberadas por el fabricante para obtener feedback de los usuarios.

- Caja Blanca: Para su realización es necesario contar con el código del programa. Si se realiza completa teniendo en cuenta todos los caminos posibles del código puede ser inaplicable, por lo tanto una variante puede ser probar los caminos más probables o frecuentes del código (que son minoría).

- Inspección: También se necesita el código del programa y consiste en leerlo de forma secuencial entendiendo lo que se escribió. Permite detectar hasta un 90% de los errores. Este proceso es menos costoso que prueba y error.

- Prueba y error: Consiste en compilar, ejecutar el programa y ver qué sucede. Sí se produce un error, el mismo debe ser corregido. Sí luego de varias compilaciones el error persiste se debe proceder con la inspección del código. Además si después de un número excesivo de compilaciones y habiendo pasado por la inspección no se puede corregir el problema se debe descartar el código y comenzar de cero.

2.C) Un motivo importante por el cual es conveniente realizar la prueba antes de la etapa de construcción del software es que no se interrumpe el flujo de trabajo al terminar la solución dado que no se debe realizar dicha prueba al final.

Otro motivo es que esta práctica permite complementar la etapa de análisis desvelando detalles sutiles sobre el problema que luego se van a tener en cuenta en la etapa de codificación. Además, permite escribir casos de prueba abstrayéndose del código sin fijarse en cómo se implementa la solución.