**PRUEBAS REALIZADAS**

**SISTEMA DE AISLAMIENTO DE PROCESOS (SANDBOX)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Prueba** | **Descripción** | **Res. Esperado** | **Veredicto** |
| C | test01.c | Programa vacío | OK | Superada |
| test02.c | Imprimir a STDOUT utilizando *write* | OK | Superada |
| test03.c | Tratar de abrir un archivo en modo lectura | RF | Superada |
| test04.c | Reservar memoria utilizando *malloc* | OK | Superada |
| test05.c | Imprimir a STDOUT utilizando *printf* | OK | Superada |
| test06.c | Verifica que una función de constructor no pueda ejecutar operaciones con privilegios | RF | Superada |
| test07.c | Lee un dato utilizando *scanf* | OK | Superada |
| test08.c | Verifica que un programa pueda salir utilizando *exit*. El código de salida tiene que ser 0 (OK), ya que si no arrojará un AT (Abnormal Termination). | AT | Superada |
| test09.c | Prueba que una función destructor funcione | OK | Superada |
| test10.c | Utilizando fork para crear hilos. | RF | Superada |
| test11.c | Ejecución de un Loop infinito | TL | Superada |
| test12.c | Imprimir en un loop infinito un mensaje | OL | Superada |
| test13.c | Creación de un arreglo muy grande utilizando con *malloc* | ML | Superada |
| test14.c | Creación de un arreglo muy grande utilizando el stack | ML | Superada |
| test15.c | Imprimir a STDERR utilizando *fputs y fprinf* | RF | Superada |
| test16.c | Escritura de archivo con *fprintf* | RF | Superada |
| test17.c | Creación de sockets para un servidor | RF | Superada |
| test18.c | División por cero | RT | Superada |
| test19.c | Uso de la función *system* para ejecutar programas externos | RT | Superada |
| Test20.java | Llamadas recursivas infinitas para verificar memoria del stack. | ML | Superada |
| C++ | test50.cpp | Usando *cout* para imprimir en STDOUT | OK | Superada |
| test51.cpp | Uso de un constructor para imprimir en STDOUT | OK | Superada |
| test52.cpp | Privilegios en constructor | RF | Superada |
| test53.cpp | Uso de destructor para imprimir en STDOUT | OK | Superada |
| test54.cpp | Verifica si los operadores *new* y *new[]* funcionan | OK | Superada |
| test55.cpp | Uso de la función *system* para ejecutar programas externos | RF | Superada |
| test56.cpp | Uso de *cin* para la lectura de archivos | OK | Superada |
| test57.cpp | Ejecución de un Loop infinito | TL | Superada |
| test58.cpp | Creación de un arreglo muy grande utilizando con *new* | ML | Superada |
| test59.cpp | Imprimir a STDERR utilizando cerr | OK | Superada |
| test60.cpp | Cliente con sockets | RF | Superada |
| JAVA | Test01.java | Programa vacío | OK | Superada |
| Test02.java | Imprimir a STDOUT utilizando print, *println, append, write, printf* | OK | Superada |
| Test03.java | Utilizando STDIN con objeto *scanner* | OK | Superada |
| Test04.java | Utilizando STDIN con objeto *bufferedReader* | OK | Superada |
| Test05.java | Clase sin método *main* | RT | Superada |
| Test06.java | Clase con método *main* pero no *static* | RT | Superada |
| Test07.java | Tratar de escribir un archivo con *PrintWriter* | RF | Superada |
| Test08.java | Creación de Hilos | RF | Superada |
| Test09.java | Reservar memoria utilizando *new* para un arreglo. Suma de enteros. | OK | Superada |
| Test10.java | Uso de la función *system* para ejecutar programas externos | RF | Superada |
| Test11.java | Loop infinito | TL | Superada |
| Test12.java | Imprimir en un loop infinito un mensaje | OL | Superada |
| Test13.java | División por cero. | RT | Superada |
| Test14.java | Creación de un servidor socket | RF | Superada |
| Test15.java | Creación de un socket para consultar pagina | RF | Superada |
| Test16.java | Creación de un arreglo muy grande utilizando con *new* | ML | Superada |
| Test17.java | Método *main static* pero no con los parámetros adecuados | RT | Superada |
| Test18.java | Llamadas recursivas infinitas para verificar memoria del stack. | ML | Superada |
| Test19.java | Tratar de invocar un objeto en awt, JOptionPane | RF | Superada |
| Test20.java | Verifica que un programa pueda salir utilizando *exit*. El código de salida tiene que ser 0 (OK), ya que si no arrojará un AT (Abnormal Termination). | AT | Superada |

NOTAS:

* JAVA: El uso de *Scanner* y *prinf* utilizan locate que necesita varios permisos potencialmente peligrosos. Posteriormente hay que ver cómo se puede corregir.
* JAVA, C\_C++: Los parámetros de los sistemas de asilamiento esperan siempre una entrada correcta. No se garantiza el funcionamiento si esto no sucede.