

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

Grado en Ingeniería Informática

PROYECTO FINAL: PRIMERA PARTE

METODOLOGÍA DE LA PROGRAMACIÓN

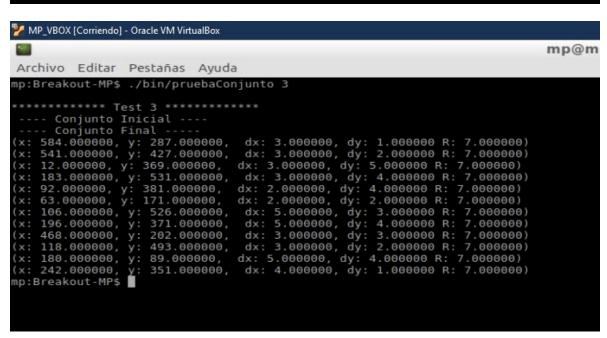
Autores:

COLLADO ROJAS, ALEXANDER - PA0274606 VIDICAN, MIHNEA IOAN - X7527819B

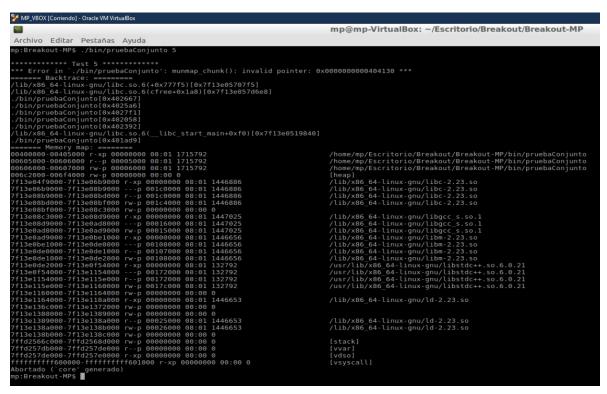
Curso: 2021-2022

1. Desarrollo de la práctica

Esta primera parte de la práctica no ha resultado ser demasiado compleja. Sin embargo, hemos obtenido un error cuando el programa intentaba liberar memoria dinámica. Por el resultado obtenido a través de las estadísticas de Valgrind, hemos pensado en un primer momento que se trataba de un fallo en el código del constructor con parámetros. Posteriormente, analizándolo con detenimiento, nos percatamos de que el fallo provenía de no inicializar el dato miembro Particula *set a 0 dentro del archivo ConjuntoParticulas.h. Puesto que hemos puesto como condición en la función que libera memoria que el dato miembro set no tenga un valor nulo, la comparación resultaba errónea. Aparte de esto, hemos obtenido algunos errores de sintaxis.



```
### Weylion | Content | Minimality | Content |
```



Por otra parte, al intentar compilar el proyecto mediante un Makefile, hemos obtenido este otro error:

```
src/ConjuntoParticulas.cpp:5:7: error: expected nested-name-specifier before 'namespace'
5 | using namespace std;
```

Esto se ha debido a no compilar con el flag -std=c++14. Pese a añadirlo en el Makefile, el fallo siguió en la máquina virtual en la que lo probamos y solo a uno de nosotros le ha funcionado compilar el programa completo.

2. Resultados finales

Finalmente, tras la solucionar todos los fallos, el resultado al ejecutar los test ha sido:

```
mp:Breakout-MP$ valgrind --leak-check=full --track-origins=yes ./bin/pruebaConjunto 6
    -3750== Memcheck, a memory error detector
-3750== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
-3750== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
    =3750== Command: ./bin/pruebaConjunto 6
   =3750==
  Prueba Constructores
 Conjunto Inicial con 5 particulas
(x: 488.000000, y: 409.000000, dx: 2.000000, dy: 4.000000 R: 7.000000)
(x: 189.000000, y: 385.000000, dx: 4.000000, dy: 2.0000000 R: 7.000000)
(x: 555.0000000, y: 200.000000, dx: 3.000000, dy: 1.0000000 R: 7.000000)
(x: 477.000000, y: 169.000000, dx: 5.000000, dy: 3.000000 R: 7.000000)
(x: 227.000000, y: 187.000000, dx: 5.000000, dy: 5.000000 R: 7.000000)
Utiles -> 5
Capacidad -> 5
Capacidad -> 5
---- Después de reemplazar particulas ----
(x: 5.000000, y: 5.000000, dx: 0.000000, dy: 0.000000 R: 5.000000)
(x: 5.000000, y: 5.000000, dx: 0.000000, dy: 0.000000 R: 5.000000)
(x: 5.000000, y: 5.000000, dx: 0.000000, dy: 0.000000 R: 5.000000)
(x: 5.000000, y: 5.000000, dx: 0.000000, dy: 0.000000 R: 5.000000)
(x: 5.000000, y: 5.000000, dx: 0.000000, dy: 0.000000 R: 5.000000)
Utiles -> 5
 Tapacidad -> 5
****** FIN
  Utiles -> 0
Capacidad -> 0
         -- Agrego 12 particulas ----
             - Conjunto Final
--- Conjunto Final ---- (x: 41.000000, y: 130.000000, dx: 2.000000, dy: 3.000000 R: 7.000000) (x: 107.000000, y: 84.000000, dx: 5.000000, dy: 5.000000 R: 7.000000) (x: 329.000000, y: 472.000000, dx: 3.000000, dy: 5.000000 R: 7.000000) (x: 304.000000, y: 420.000000, dx: 1.000000, dy: 4.000000 R: 7.000000) (x: 588.000000, y: 57.000000, dx: 4.000000, dy: 2.000000 R: 7.000000) (x: 228.000000, y: 566.000000, dx: 5.000000, dy: 2.000000 R: 7.000000) (x: 233.000000, y: 552.000000, dx: 3.000000, dy: 4.000000 R: 7.000000) (x: 291.000000, y: 300.000000, dx: 3.000000, dy: 1.000000 R: 7.000000) (x: 245.000000, y: 391.000000, dx: 3.000000, dy: 5.000000 R: 7.000000) (x: 555.000000, y: 587.000000, dx: 5.000000, dy: 3.000000 R: 7.000000) (x: 82.000000, y: 131.000000, dx: 4.000000, dy: 5.000000 R: 7.000000) (x: 461.000000, y: 364.000000, dx: 5.000000, dy: 2.000000 R: 7.000000) Utiles -> 12
  tiles -> 12
apacidad -> 12
*****
                                               FIN *********
```

```
******* INICIO ********
       - Conjunto Inicial con 11 particulas al azar ----
300.000000, y: 197.000000, dx: 5.000000, dy: 4.000000 R: 7.000000)
     300.000000, y: 197.000000, dx: 5.000000, dy: 4.000000 R: 7.000000)
14.000000, y: 469.000000, dx: 1.000000, dy: 1.000000 R: 7.000000)
327.000000, y: 267.000000, dx: 5.000000, dy: 1.000000 R: 7.000000)
491.000000, y: 285.000000, dx: 2.000000, dy: 3.000000 R: 7.000000)
337.000000, y: 308.000000, dx: 1.000000, dy: 2.000000 R: 7.000000)
380.000000, y: 19.000000, dx: 3.000000, dy: 3.000000 R: 7.000000)
149.000000, y: 173.000000, dx: 5.000000, dy: 5.000000 R: 7.000000)
537.000000, y: 411.000000, dx: 3.000000, dy: 3.000000 R: 7.000000)
7.000000, y: 502.000000, dx: 4.000000, dy: 3.000000 R: 7.000000)
122.000000, y: 56.000000, dx: 5.000000, dy: 5.000000 R: 7.000000)
322.000000, y: 105.000000, dx: 5.000000, dy: 2.000000 R: 7.000000)
les -> 11
Utiles -> 11
Capacidad -> 11
  ---- Elimino todas las particulas en orden aleatorio ----
     -- Conjunto Final
Utiles -> 0
Capacidad -> 2
    .
*********
                              FIN
                                          ******
        ******* TEST 5 ******
   ********** INICIO ********
   --- Conjunto inicial con 10 particulas -----
  ---- Agrego/Borro particulas al azar
                             FIN
 =3750==
 =3750== HEAP SUMMARY:
                     in use at exit: 72,704 bytes in 1 blocks
total heap usage: 334 allocs, 333 frees, 127,979 bytes allocated
  =3750==
  =3750==
 =3750== LEAK SUMMARY:
  =3750==
                       definitely lost: 0 bytes in 0 blocks
                       indirectly lost: 0 bytes in 0 blocks
possibly lost: 0 bytes in 0 blocks
still reachable: 72,704 bytes in 1 blocks
suppressed: 0 bytes in 0 blocks
  =3750==
  =3750==
  =3750==
  =3750==
  =3750== Reachable blocks (those to which a pointer was found) are not shown.
=3750== To see them, rerun with: --leak-check=full --show-leak-kinds=all
  =3750==
  =3750== For counts of detected and suppressed errors, rerun with: -v
   3750== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
  p:Breakout-MP$
```

TEST	EJECUTA	SALIDA CORRECTA	VALGRIND
T ₁	SÍ	SÍ	BIEN
T ₂	SÍ	SÍ	BIEN
Тз	SÍ	SÍ	BIEN
T4	SÍ	SÍ	BIEN
T ₅	SÍ	SÍ	BIEN

A su vez, la instalación y ejecución de Raylib junto con los ficheros de ejemplos han resultado fáciles, aunque por ahora no hemos hecho ningún desafío preliminar. Conforme vayamos avanzando el desarrollo de la práctica, informaremos de sugerencias y otras pruebas realizadas.