Universidad Central de Venezuela

Facultad de Ciencias

Escuela de Computación

Algoritmos y Programación

ANÁLISIS DEL PROYECTO N° 1

Estudiantes:

César Carios

Jhonatan Homsany

Sección C3

Para llegar a la solución final de este proyecto se consideró conveniente dividir el mismo en cinco fases de desarrollo distintas, de esta forma se tuvo una estructura de trabajo organizada que ayudó a resolver los problemas que se presentaban con más facilidad. Las cinco fases en las que se dividió el proyecto fueron las siguientes:

Fase 1: Trabajar con las variables de Mario y sus copias.

Fase 2: Trabajar con los Scuttlebugs.

Fase 3: Trabajar con la estrella y sus funciones.

Fase 4: Trabajar en las iteraciones del programa.

Fase 5: Trabajar en las validaciones de las entradas del programa.

Durante el desarrollo de la fase 1 hasta la fase 4 se asumió que las entradas eran correctas, luego en la fase 5 se hicieron las pruebas con entradas incorrectas para verificar que el programa sabía reaccionar ante ellas. A continuación se describirá qué se hizo en cada una de las fases:

Fase 1: En esta fase de desarrollo se dieron los primeros pasos para comenzar el proyecto: declarar las variables de las posiciones iniciales de Mario y sus copias. Luego, declarar las variables que representarían los sentidos de Mario y de sus copias además de una variable a la que se llamó "velocidad" (distancia que recorrería Mario en cada iteración). Esta variable "velocidad" solo se usó en la fase 1 para hacer pruebas, posteriormente se eliminó y se añadió en la fase 4. También se declaró una variable llamada "cuadrante", cuyo objetivo fue establecer en qué cuadrante del plano cartesiano estaría Mario dependiendo de sus coordenadas.

A continuación, se procedió a igualar las posiciones iniciales y la velocidad de las copias de Mario con la posición inicial y la velocidad del Mario original, de esta forma todos empezarían en las mismas coordenadas y tendrían la misma velocidad. Ahora se trabajaría con la variable llamada "cuadrante". Usando condicionales "if" y "else if" se logró definir en qué cuadrante estaría Mario en función de sus coordenadas. Dependiendo de las coordenadas de Mario, la variable "cuadrante" adoptaría el valor de 1, 2, 3 o 4 para representar en qué parte del plano cartesiano estaba Mario. Teniendo esto definido, se pudo continuar y validar qué sucedería con el sentido de las copias de Mario

dependiendo de en qué cuadrante estaría el Mario original. Se usó el condicional switch en función de la variable "cuadrante". Así se pudo definir en qué cuadrantes cambiaban los sentidos de las copias de Mario, y si cambiaban en el eje de las ordenadas o de las abscisas o en ambas.

Fase 2: Después de dar como concluida la fase 1, se empezó la fase 2. La fase 2 tiene que ver con los Scuttlebugs y sus funciones. Lo primero que se hizo en esta fase fue declarar la variable que guardaría la cantidad de Scuttlebugs que el usuario introduciría al programa (máximo 3). Luego, se declararon las variables que representarían las posiciones de los Scuttlebugs en el plano cartesiano. Y las últimas variables que se declararon fueron las que guardarían los puntos de salud de cada Scuttlebug del programa. El paso siguiente fue definir qué sucedería si la posición de Mario o una de sus copias coincidía con la posición de un Scuttlebug. En este caso, Mario le quitaría dos puntos de vida, y sus copias un solo punto de vida si coincidían en el plano cartesiano con algún Scuttlebug. Para expresar esto en código, se usó un condicional switch en función de la variable "CScuttlebugs" (contiene la cantidad de Scuttlebugs). Si la cantidad de Scuttlebugs era 1, el caso 1 tendría un algoritmo que definiría la resta de los puntos de salud al Scuttlebug si coincidía con alguna copia de Mario o Mario original. Los demás casos del switch contienen un algoritmo similar, solo varía la cantidad de Scuttlebugs y las especificaciones para cada copia de Mario o el Mario original si coincidía en el plano cartesiano con alguno de los enemigos. Es importante destacar que en los casos del switch descrito también hay un algoritmo que define qué pasa si Mario o una de sus copias obtienen el efecto de la estrella y se encuentra con un enemigo; esto será abordado en la descripción de la fase 3. Otro detalle especificado dentro del switch descrito en este párrafo es que si Mario o una de sus copias eliminan a un Scuttlebug, Mario y sus copias se alinean en las coordenadas del enemigo abatido, para hacer esto se utilizaron condicionales "if" y "else if".

Fase 3: Esta fase abarca la declaración de las variables que contienen las coordenadas del plano cartesiano en el cual se ubica la estrella y la declaración de una variable a la que se llamó "DStar", cuyo funcionamiento fue establecer que si Mario o una de sus copias obtenía el power-up de la estrella, que el mismo les durara solo por dos iteraciones. Si Mario o una de sus copias obtenían el

power-up de la estrella y se encontraban con algún Scuttlebug, la vida del enemigo se reduciría automáticamente a cero, independientemente de la vida que tuviera al inicio del programa. Para traducir esto al código, se usó la variable "DStar" en un condicional "if" dentro del switch hecho en la fase 2 del desarrollo del programa. Esto se hizo para cada uno de los casos del switch. Ahora bien, al principio de esta fase se aclaró que el efecto de la estrella en Mario o en algunas de sus copias tenía que durar solo por dos iteraciones. Se explicará cómo se hizo esto en código en la descripción de la fase 4.

Fase 4: Hasta la fase 3, el programa funcionaba correctamente. Sin embargo, faltaba desarrollar lo que se consideró el paso más importante: las iteraciones del programa. Para esto se declararon siete variables, una llamada "NIteraciones", que guarda el número de iteraciones que el usuario introduzca y 6 variables que guardan la coordenadas finales de Mario y sus copias al terminar una iteración. Estas variables fueron igualadas a las de las posiciones de Mario y sus copias al terminar una iteración, de esta forma las coordenadas finales al concluir una iteración se convertirían en las coordenadas iniciales al iniciar una nueva iteración. Después de declarar las variables mencionadas anteriormente, se usó un bucle "for" para definir el número de veces que se ejecutaría el programa. Con ayuda de una variable "i" inicializada en cero, se declaró que todo el código que está dentro del "for" se repitiera hasta que alcanzara el número de iteraciones introducidas por el usuario.

En la descripción de la fase 3 se mencionó que la duración del power-up de la estrella en Mario o en algunas de sus copias tenía que durar solo dos iteraciones. En esta fase se definió la forma en la que esa duración se traduciría al código: con ayuda de un condicional "if" dentro del bucle "for", se definió que si la variable "DStar" es mayor a 0, entonces a esa variable se le restaría una unidad. Ese trozo de código estaría apoyado por otro condicional "if" cuya función fue igualar "DStar" a 3 si Mario o algunas de sus copias coincidían con la posición de la estrella. Estos condicionales al trabajar en conjunto darían como resultado que la duración del power-up de la estrella en Mario o sus copias duraría solo dos iteraciones. Profundizando más en el funcionamiento del efecto de la estrella es importante hacer énfasis en que el código primero evalúa si Mario o alguna de sus copias obtienen el power-up. Si esto es cierto, la duración del efecto de la estrella ("DStar") sube a 3 iteraciones. Esto se hizo así para que en la siguiente iteración después de que alguna copia o Mario obtengan la estrella,

a ese 3 de la duración se le restara 1. Así quedaría “DStar = 2”, que significa que la duración del power-up tendrá dos iteraciones. Se podría resumir de esta manera: la duración del power-up empieza a ser 2 en la iteración siguiente en la que Mario o alguna de sus copias coinciden con las coordenadas de la estrella.

Fase 5: Para empezar a trabajar en la fase 5, se asumió que las entradas siempre serían número enteros. Dicho esto, solo se tuvo que validar que las entradas para ciertas variables cumplieran con las condiciones necesarias para poder ejecutar el código de forma exitosa. Primero se declaró una variable llamada “ErrorEnLaEntrada” y se inicializó en “false” (es una variable de tipo booleano). El propósito de esta variable fue que si se volvía “true” significaba que había un error en la entrada y esto provocaría que el programa tuviera como salida un texto que decía “Error en la entrada”, y eso pasaba con ayuda de un condicional “if”. Por ejemplo, si el usuario introduce un número diferente de 1 o -1 cuando se le pide el sentido de Mario o alguna de las copias (ya sea en el eje de las abscisas o de las ordenadas), la variable “ErrorEnLaEntrada” se convertiría en “true”. Por lo tanto arrojaría un “Error en la entrada” en la salida del programa. Se hizo el mismo procedimiento para validar que la cantidad de Scuttlebugs en la simulación fueran máximo 3 y mínimo 0, y también para validar la cantidad de iteraciones del programa (mayor a 0). Fuera del bucle “for” que se describió en la fase 4, se utilizó un condicional para establecer cuándo en la salida tenía que imprimirse “Perfectamente alineado”, seguido de la posición de Mario original (que sería también la posición de las copias) y la posición de los Scuttlebugs que quedaran vivos y sus respectivos puntos de vida. Si lo anterior no se cumplía, un condicional “else if” entraría en acción y se imprimiría en la salida “Desincronización total”, seguido de la posición del Mario original y la posición de las demás copias en el plano cartesiano al terminar el programa.