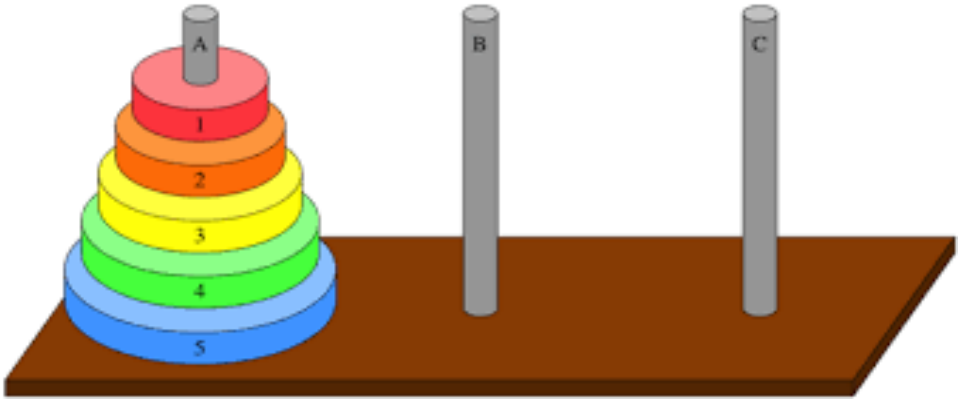
UNIVERSIDAD DE FUERZAS ARMADAS “ESPE”

MATERIA:
ESTRUCTURA DE DATOS
3688
GRUPO 5



Torres de Hanoi

Manual

Versión: 001

Fecha:11 /06/2021

HOJA DE CONTROL

Organismo	ESPE		
Proyecto	TORRES DE HANOI		
Entregable	Manual de Usuario		
Autor	ESPE-ING EN SOFTWARE		
Versión/Edición	0100	**Fecha Versión**	7/06/2021
Aprobado por		**Fecha Aprobación**	11/06/2021
		Nº Total de Páginas	11

DESARROLLADORES

Nombre y Apellidos
Jurado Junior
Paguay Alex
Román Yulliana
Rosero Theo
Sañay Santiago

ÍNDICE

1 DESCRIPCIÓN DEL SISTEMA

1.1 Objetivo

Simular las torres de Hanoi en una interfaz gráfica, su finalidad es mover todos los discos de la torre de la izquierda a la torre de la derecha. La torre de en medio es para almacenamiento temporal de las piezas

1.2 Alcance

La presente simulación está dirigida a los alumnos de la Asignatura de Estructura de Datos de la Carrera de Ingeniería de Software de la UFA ESPE

1.3 Funcionalidad

La simulación de las Torres de Hanoi permite visualizar en interfaz gráfica, el juego de 4 hasta 8 discos con 3 torres, evitando que se apilen discos más grandes sobre discos más pequeños. Además, la simulación permitirá visualizar los movimientos necesarios para trasladar toda una pila a otra torre.

2 DESCRIPCIÓN DEL SISTEMA

El sistema muestra un mensaje solicitando al usuario ingresar el número de discos que desea que tenga la torre, posteriormente muestra a las tres torres con discos de distintos tamaños y colores, seguido se presenta como los discos se van intercambiando hasta llegar a la torre del lado derecho. De igual forma, muestra la lista de pasos que se necesitaron para obtener la solución.

2.1 Clase 1 (Disk)

La clase disk se encarga de graficar los discos de la torre usando funciones y tipos de datos abstractos de la librería gráfica SFML que ayudan a que dichos discos tengan un color y tamaño propio. Usando las proporciones del disco más grande se posicionan los discos más pequeños de forma que queden centrados.

2.1.1 Pantalla

```
1   #pragma once
2   #include <iostream>
3   #include <SFML/Graphics.hpp>
4   class Disk
5   {
6   private:
7       sf::RectangleShape _disk;
8       sf::Vector2f _position = sf::Vector2f(0, 0);
9       sf::Color _color;
10      sf::Vector2f _size;
11
12  public:
13      Disk(sf::Vector2f);
14      void draw_disk(sf::RenderWindow*);
15      void set_color(sf::Color);
16      void set_position(sf::Vector2f);
17      sf::Vector2f get_position();
18      sf::Vector2f get_size();
19  };
20
```

2.1.2 Descripción de los atributos

- **sf::RectangleShape _disk;**
  Tipo de dato para representar un rectángulo.

- **sf::Vector2f _position = sf::Vector2f(0, 0);**

Tipo de dato abstracto, que muestra la coordenada inicial desde la que se va a dibujar.

- **sf::Color _color;**

Este atributo define el color de los discos, se puede asignar a partir de un entero sin signo en el rango rgb[0, 255].

- **sf::Vector2f _size;**

Este atributo define el tamaño de los discos ancho y alto.

2.1.3 Descripción de los métodos

- **Disk(sf::Vector2f);**

Constructor de la clase **"Disk"** recibe de parámetros la posición.

- **void draw_disk(sf::RenderWindow*);**

Dibuja el disco en la ventana de la aplicación además recibe como parámetro la dirección de la ventana en donde se va ha realizar el dibujo. .

- **void set_color(sf::Color);**

Permite colorear las figuras recibiendo de parámetro un tipo de dato Color

- **void set_position(sf::Vector2f);**

Setter de la posición del disco.

- **sf::Vector2f get_position();**

Getter de la posición del disco.

- **sf::Vector2f get_size();**

Getter del tamaño del disco

3 Clase 2 (Tower)

La clase tower se encarga de dibujar las torres donde van a ser apilados los discos, las torres se grafican con ayuda de rectángulos, con la librería SFML RectangleShape recibe las posiciones en pantalla con el tipo Vector2f y se dibuja en pantalla.

3.1 Pantalla

```
1   #pragma once
2   #include <vector>
3   #include <SFML/Graphics.hpp>
4   #include "Disk.h"
5   class Tower
6   {
7   private:
8       std::vector<Disk*> _disks;
9       sf::RectangleShape _base;
10      sf::RectangleShape _tower;
11      sf::Vector2f _position = sf::Vector2f(0, 0);
12
13  public:
14      Tower();
15      void draw_structure(sf::RenderWindow*);
16      void add_disk(Disk*);
17      void set_position(sf::Vector2f);
18      Disk* get_disk();
19      sf::Vector2f get_size();
20  };
21
```

3.2 Descripción de los Atributos

- **std::vector<Disk*> _disks;**

Este atributo representa a un vector de tipo Disk.

- **sf::RectangleShape _base;**

Este dato dibuja la base rectangular horizontal de la torre.

- **sf::RectangleShape _tower;**

Este dato dibuja la base rectangular vertical de la torre.

- `sf::Vector2f _position = sf::Vector2f(0, 0)`

  Tipo de dato abstracto, que muestra la coordenada inicial desde la que se va a dibujar.

3.3 Descripción de los métodos.

- `Tower();`

  Constructor de la clase Tower, este constructor no recibe parámetros.

- `void draw_structure(sf::RenderWindow*);`

Función que dibuja la estructura de la torre, la base horizontal y la base vertical. Tiene como parámetro la ventana en la que se va a graficar.

- `void add_disk(Disk*);`

  Añade un disco a la torre, tiene como parámetro un puntero de tipo Disk. Para añadir un elemento a la función push_back.

- `void set_position(sf::Vector2f);`

Esta función modifica la posición de la figura, tiene como parámetro un tipo de dato abstracto de la librería SFML.

- `Disk* get_disk();`

  La función devuelve un puntero de tipo Disk.

- `sf::Vector2f get_size();`

  Esta función devuelve el tamaño de la torre.

4 Clase 3 (App)

La clase se encarga de instanciar los elementos de tipo Tower y Disk además se encarga de ejecutar el algoritmo de las torres de Hanoi el cual comprueba si se puede mover un disco a otra torre comprobando si no existe uno más grande que este.

4.1 Pantalla 1

```
1   #pragma once
2   #include <SFML/Graphics.hpp>
3   #include "Tower.h"
4   #include "Disk.h"
5   #include <vector>
6   #include <chrono>
7   #include <random>
8   class App
9   {
10  private:
11      sf::RenderWindow *window;
12      Tower *_tower_a, *_tower_b, *_tower_c;
13      std::vector<std::vector<Tower *>> _move_list;
14
15  public:
16      App() = default;
17      App(sf::RenderWindow *);
18      void create_element();
19      void app_draw(sf::Time &timer);
20      void hanoi(int, Tower *, Tower *, Tower *);
21      int random_int(int min, int max);
22  };
23
```

4.2 Descripción de los atributos

- **sf::RenderWindow *window;**

Tipo de dato abstracto de la librería que define la ventana en la que se va a graficar.

- **Tower *_tower_a, *_tower_b, *_tower_c;**

Punteros de tipo Tower, que representan a la torre de la derecha, de la izquierda y de en medio.

- **std::vector<std::vector<Tower *>> _move_list;**

Este atributo guarda los pasos que se realizan para llegar a la solución.

4.3 Descripción de los métodos

- **App() = default;**

  Constructor por defecto de la clase App.

- **App(sf::RenderWindow *);**

  Constructor que recibe como parámetro la ventana en la que se va a dibujar.

- **void create_element();**

  Esta función se encarga de crear las torres y los discos, obteniendo su posición, color y tamaño.

- **void app_draw(sf::Time &timer);**

  Dibuja la estructura de la torre, y realiza los cambios de los discos de la torre izquierda a la derecha.

- **void hanoi(int, Tower *, Tower *, Tower *);**

  Registra los movimientos que se realizan para que los discos pasen a la torre de la derecha.

- **int random_int(int *min*, int *max*);**

Función que se utiliza para obtener el color del disco al azar, a partir de un número randómico que va entre el 0 al 255.

BIBLIOGRAFÍA Y REFERENCIAS

Referencia	Título
.....https://www.sfml-dev.org/	.....SFML (Descarga de libreria grafica)