



USAC

TRICENTENARIA

Universidad de San Carlos de Guatemala

INGENIERIA

CUNOC

Laboratorio de Estructura de Datos

Proyecto 1: Documentación técnica

Sección: A

Nombre:

Registro académico:

Mariano Francisco Camposeco Camposeco

202030987

Quetzaltenango, 03 de abril de 2022.

Introducción

En esta documentación encontrará que funcionalidad tiene alguna clase, método, encabezado creado, para asimilar del cómo puede llegar a funcionar determinada instrucción, siendo entre lenguaje de alto y bajo nivel, con el fin de que no sea ni complejo ni tan sencillo que deje huecos en las explicaciones, así mismo contando con la lógica del funcionamiento del programa.

OBJETIVOS

Objetivo general:

Explicar las diferentes funciones realizadas en el código.

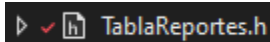
Objetivos específicos:

1. Entender el qué funcionalidad se tiene en las matrices ortogonales.
2. Aprender para qué se realizó un método.
3. Asimilar la lógica en el programa.

Lógica del juego del 15 estructurado:

El juego clásico del 15 consiste en tener determinada cantidad de bloques del 1-15 y con un bloque vacío, donde nuestra meta es ordenar de manera que sea ascendente. Ahora bien, en este juego del 15, se cuenta con la capacidad de tener un tablero dinámico, donde se pueda crear tableros 2x2,3x5,6x6, como se brinde el gusto siempre y cuando la fila y columna sean mayores a 1, así mismo, se puede crear distintos niveles, como mínimo 1 y se irá moviéndose entre niveles intercambiando el espacio vacío con el valor del otro nivel.

Encabezado tabla reportes:



Clase tablaReporte:

//variables

Privado: posición,nombre,punteo,tiempo

//Nos sirve para inicializar valores

Publico: constructor TablaReporte()

//Utilizado para crear la tabla con los valores obtenidos

construirTabla(int, string, int, float)

//llamar valores

getPosicionReporte()

getPunteoReporte()

getTiempoPartida()

//modificar valores

setPosicionReporte(int)

//imprimir datos

mostrarTabla()

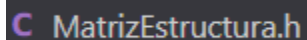
//Intercambiar valores de tabla

swap(TablaReporte& valor1, TablaReporte& valor2)

//hacemos uso del método de ordenamiento QuickSort, para posiciones

ordenarTabla(vector<TablaReporte>, int, int)

Encabezado matrizEstructura :



struct crearNodo: //Como necesitamos matriz ortogonal usamos nodos

//variables

crearNodo* up, down, left, right;

//constructor

crearNodo(int valorData, int valorX, int valorY) ;

Struct DatoVertical, Struct DatoHorizontal: //Creamos los datos verticales y horizontales

//variables

crearNodo* start, end;

//ingresamos el siguiente dato

void ingresarDatoAdelante(crearNodo* ingresado) ;

//Ingresamos dato al final

void ingresarDatoEnd(crearNodo* ingresado)

//Ingresamos dato a la mitad

void ingresarDatoMitad(crearNodo* ingresado)

//Verificamos si está vacío

bool emptyDato()

//Los nodos superiores y laterales

struct nodoEncabezado, struct encabezados, struct nodoLado, struct lados

//ingresamos el siguiente dato

void ingresarDatoAdelante() ;

//Ingresamos dato al final

void ingresarDatoEnd()

//Ingresamos dato a la mitad

void ingresarDatoMitad()

//Verificamos si está vacío

bool emptyDato()

//Creamos matriz

struct matrizDinamica

//Le pasamos datos a ingresar

void ingresar(int valorX, int valorY, int ingresado)

//llenar nuestros datos manualmente

void completarManual(int valorX, int valorY)

//llenar nuestros datos aleatorios

void completarManual(int valorX, int valorY)

//llenamos tableros con archivo de entrada

void completarTableroArchivo(int valorX, int valorY, vector<int> valoresTableroArchivo)


//imprimir nuestros datos

```

void imprimir(int valorX)
//verificacion de punteo
vector<int> pasarValoresPunteo(int valorX)
//Intercambio hacia arriba
int intercambioVacioArriba(int valorX)
//intercambio hacia abajo
int intercambioVacioAbajo(int valorX)
//Intercambio hacia la izquierda
int intercambioVacioIzquierda(int valorX)
//intercambio hacia la derecha
int intercambioVacioDerecha(int valorX)
//llenar nuestros datos manualmente para nivel 2 en adelante
void completarManualNivelesMas(int valorX, int valorY, int contadorNumerosAleatorios)
//llenar nuestros datos aleatoriamente para nivel 2 en adelante
void completarAleatorioNivelesMas(int valorX, int valorY, int contadorNumerosAleatorios)
//buscar posicion donde se encuentra el cero
int posicionCero(int valorX)
//cantidad de movimientos para encontrar el cero
int cantidadMovimientosCero(int valorX)
//buscar posicion donde se encuentra el cero
int recuperarValorOtroNivel(int valorI, int contadorMovimientos)
//actualizar a valor del otro nivel donde estaba el bloque vacio
void actualizamosNivelAnterior(int valorI, int contadorMovimientos, int valorRecuperado)
//actualizar a cero la posicion enlazada
void actualizamosNivelSiguiente(int valorI, int contadorMovimientos)

```

Proyecto1-EstructuraDeDatos:

 Proyecto1-EstructuraDeDatos.cpp

```

//creacion de variables
Linea 13-22
//creamos métodos
Linea 24-34
//creamos datos con matriz dinámica
Linea 35-38
/*metodos usados
*/creamos nuestro menú con todas las opciones posibles
void menu();

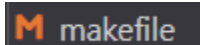
```

```

//llamamos al reporte generado de ultimo
void reportes();
//vemos los resultados obtenidos durante la partida
void tablaResultados();
//indicamos las opciones que se tienen para jugar
void opcionesJugar();
//entramos a jugar ya con los tableros creados
void jugando();
//leemos archivo
void leerArchivo();
//buscamos el valor de cero, si se encuentra en determinada fila
void localizarValorCero();
/*matrices
*/llenamos matriz aleatoriamente
void llenarMatrizAleatoria();
//llenamos matriz manualmente
void llenarMatrizManual();
//llenamos matriz aleatoria si hubiera más de un nivel
void llenarMatrizAleatoriaNivelesMas();
//llenamos matriz manual si hubiera más de un nivel
void llenarMatrizManualNivelesMas();
//llenamos tablero por medio de un archivo
void llenarTableroArchivo(vector<int> valoresTableroArchivo);

```

MakeFile:



```

//Definimos nombre de .exe
Juego15
//Para mostrar las advertencias
-Wall
//Para limpiar archivos
clean
//Para crear .exe en comando
Make
//Para eliminar en comando
Make clean

```