UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
SEGUNDO SEMESTRE 202

C

MANUAL TÉCNICO
FASE 2

Nombre: Angel Francisco Sique Santos

Carné: 202012039

INTRODUCCIÓN

La empresa Usac Games desea implementar un videojuego que permitan desarrollar la agilidad mental de los usuarios, por lo cual ha planeado desarrollar una aplicación con el juego Batalla naval y le solicita a usted como estudiante de estructura de datos poder implementar algoritmos, funciones y estructuras que permitan que el juego tenga un rendimiento óptimo y fluido.

REQUISITOS MÍNIMOS

- 500mb de disco duro
- 2gb de RAM
- Linux o MacOs
- Equipo Intel Pentium o superior
- Python 3.10

LIBRERÍAS

- Graphviz
- Pillow 9.2.0
- PySimpleGui 4.60.3
- Glove
- SHA256
- JsonCPP

OPCIONES DEL PROGRAMA

Primero iniciamos las estructuras que usaremos y agregamos el primer usuario que será el administrador.

```
ListaUsuarios usuarios;
ListaArticulos articulos;
ColaTutorial tutorial;
Cabecera cabecera;
ArbolB arbol;
string archivo = "";
//menu(usuarios, articulos, tutorial, cabecera);
nodoUsuarios*admin = new nodoUsuarios();
admin→nick = "EDD";
admin→password = encriptarSHA256("edd123");
admin→edad = 20;
admin→monedas = 0;
admin→id = 0;
  You, hace 6 días • Login arbol b
arbol.insertar(admin);
```

```
Servidor API(usuarios,tutorial,articulos,cabecera,archivo,arbol);
GloveHttpServer serv(8080, "", 2048);
serv.compression("gzip, deflate");
namespace ph = std::placeholders;
serv.addRest("/Cargar/$ruta", 1,
             GloveHttpServer::jsonApiErrorCall,
             std::bind(&Servidor::getUsuarios, &API, ph::_1, ph::_2),
             std::bind(&Servidor::postRuta, &API, ph::_1, ph::_2));
serv.addRest("/ObtenerUsuarios/", 0,
             GloveHttpServer::jsonApiErrorCall,
             std::bind(&Servidor::getUsuariosJson, &API, ph::_1, ph::_2));
serv.addRest("/ObtenerUsuario/$nick/$password/$id", 3,
             GloveHttpServer::jsonApiErrorCall,
             std::bind(&Servidor::getUsuario, &API, ph::_1, ph::_2));
serv.addRest("/EliminarUsuario/$id", 2,
             GloveHttpServer::jsonApiErrorCall,
             std::bind(&Servidor::postEliminarUsuario, &API, ph::_1, ph::_2));
serv.addRest("/CrearUsuario/$nick/$password/$edad/", 3,
             GloveHttpServer::jsonApiErrorCall,
```

Luego iniciamos el servidor y agregamos los endpoint que usaremos, todo esto es en c++.

Luego en python lo primero que hacemos es importar todas las librerias que usaremos.

```python
import io
import random
import string
import webbrowser
import json
import requests
from PIL import Image
import PySimpleGUI as sg
```

Después podemos empezar con el menú principal, en este caso usamos la librería PySimpleGui para hacer la interfaz gráfica.

```python
#Menu principal
def menuPrincipal():
    sg.theme('DarkTeal4')

    layout = [[sg.Text('Menu',size = (10,0), font="Arial 30 bold", justification='center')],
              [sg.Button('Cargar datos', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Crear Usuarios', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Iniciar Sesion', size = (20,0), font="Arial 15 bold")]
              ]
    window = sg.Window('Menu', layout, size=(300, 250), element_justification='c')
    while True:
        event, values = window.read()
        if event == sg.WIN_CLOSED:…
        if event == 'Cargar datos':…
        if event == 'Crear Usuarios':
            window.hide()
            registrar_usuario()
            window.un_hide()
        if event == 'Iniciar Sesion':…
    window.close()
```

Aquí tenemos las opciones de cargar el archivo de entrada, crear un usuario o iniciar sesión. Si decidimos cargar el archivo nos abrirá una ventana donde elegiremos el archivo que queramos usar, si escogemos crear usuario nos pedirá los datos del usuario y si queremos iniciar sesión solo tendremos que poner el usuario y la contraseña.

Al iniciar sesión desde python se hará un get para verificar que la contraseña y el usuario

```python
#verificar el inicio de sesion
def verify_login( nick,  password):      You, hace 2 semanas • API creada
    try:
        progress_bar()
        print(nick)
        print(password)
        print(f'{base_url}ObtenerUsuario/' + nick + '/' + password + '/' + regresar_id_usuario(nick) + '/')
        res = requests.get(f'{base_url}ObtenerUsuario/' + nick + '/' + password + '/' + regresar_id_usuario(nick) + '/')
        data = res.text#convertimos la respuesta en dict
        data = json.loads(data)
        print(data)
        usuario_global['nick'] = data['usuario'][0]['nick']
        usuario_global['password'] = data['usuario'][0]['password']
        usuario_global['monedas'] = data['usuario'][0]['monedas']
        usuario_global['edad'] = data['usuario'][0]['edad']
        usuario_global['id'] = data['usuario'][0]['id']
        return (data['status'])
```

coincidan.

En caso de que sea un inicio de sesión exitoso se verificara si el usuario es uno normal o el administrador, si es un usuario normal se mostrara un menú con opciones para iniciar un juego, ver la tienda, ver el tutorial, editar el usuario, eliminar el usuario y ver el reporte de las compras hechas.

```
#Menu
def menu():      You, hace 2 semanas • API creada
    layout = [[sg.Text('Menu',size = (10,0), font="Arial 30 bold", justification='center')],
              [sg.Button('Cerrar Sesion', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Editar Usuario', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Eliminar Usuario', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Iniciar Juego', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Iniciar Tutorial', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Tienda', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Reportes', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Salir', size = (20,0), font="Arial 15 bold")]]
    window = sg.Window('Menu', layout, size=(300, 430), element_justification='c')
    while True:
        event, values = window.read()
        if event == sg.WIN_CLOSED or event == 'Salir':…
        if event == 'Cerrar Sesion':…
        if event == 'Editar Usuario':…
        if event == 'Eliminar Usuario':…
        if event == 'Iniciar Juego':…
        if event == 'Tienda':…
        if event == 'Reportes':…
        if event == 'Iniciar Tutorial':…
    window.close()
    return event
```

Si cerramos sesión volverá a pedir los datos del nuevo usuario, si eliminamos el usuario se preguntará si estamos seguros y si la respuesta es sí lo eliminara por completo, si abrimos la tienda se nos mostraran todos los artículos disponibles, si iniciamos un juego nos preguntara el tamaño del tablero y lo creara.

```
    #Si el ancho y alto es mayor a 10
    if(tamanio >= 10):
        layout = []
        botones = []
        #Numero de barcos
        constante = int(((tamanio - 1)/10)+1)
        Portaaviones = 1*constante
        Submarino = 2*constante
        Destructor = 3*constante
        Buque = 4*constante
        vidas = 3
        layout.append(…
        matriz = MatrizDispersa()
        for i in range(0, (tamanio) + 1):…

        nombreJuego = 'Juego' + str(usuario_global['juegos'])
        usuario_global['juegos'] = str(int(usuario_global['juegos']) + 1)
        #Colocar barcos
        window =  sg.Window('Menu', layout, element_justification='c')
        while True:…
    else:…
```

Al crear el tablero se verificara que el tamaño indicado sea mayor o igual a 10. Luego calculara la cantidad de barcos que serán colocados en el tablero. Después de ser colocados se mostrara el tablero y el usuario podrá empezar a jugar.

Al seleccionar una posición se recorrerá el tablero para encontrar la posición seleccionada. Al ser encontrada se verificara que haya o no un barco en esa posición, si lo hay se sumaran los puntos y se mostrara en el tablero, de lo contrario se restaran esos puntos y se mostrara en el talero la posición seleccionada.

```
if(vidas > 0 and revisar_tablero(matriz,layout) == False):
    for k in range(1 , len(layout)):
        for l in range(len(layout) - 1):
            if(layout[k][l].ButtonText == event ):
                if(pintar_disparo(k, l, matriz, layout)):
                    agregar_movimiento(k-1,l,nombreJuego,usuario_global['id'])
                    usuario_global['monedas'] = int(usuario_global['monedas']) + 20
                    layout[0][4].update(usuario_global['monedas'])
                    actualizar_monedas(+20)
                    puntos = puntos + 20
                else:
                    vidas = vidas - 1
                    agregar_movimiento(k-1,l,nombreJuego,usuario_global['id'])
                    layout[0][1].update(vidas)
                    Errores += 1
                break
        else:
            continue
        break
else:…
```

El tutorial es parecido, primero se obtienen las posiciones a través de un get hacía c++ y luego se comenzara a explicar paso por paso los movimientos obtenidos desde c++.

```
#Obtener tutorial
def obtener_tutorial():
    """…
    try:
        res = requests.get(f'{base_url}ObtenerTutorial/')
        data = res.text#convertimos la respuesta en dict
        data = json.loads(data)
        return data
    except:
        pass
```

```
    sg.theme('DarkTeal4')
    tamanio = data['ancho']
    llenado = False
    repetir = True
    monedasTutorial = 100
    #Si el ancho y alto es mayor a 10
    if(tamanio >= 10):
        layout = []
        botones = []
        #Numero de barcos
        constante = int(((tamanio - 1)/10)+1)
        Portaaviones = 1*constante
        Submarino = 2*constante
        Destructor = 3*constante
        Buque = 4*constante
        vidas = 3
        layout.append(…
        matriz = MatrizDispersa()
        for i in range(0, (tamanio) + 1):…

        #Colocar barcos
        window =  sg.Window('Menu', layout, element_justification='c')
        sg.popup('Bienvenido al tutorial', title='Bienvenido')
```

```
    #Colocar barcos
    window =  sg.Window('Menu', layout, element_justification='c')
    sg.popup('Bienvenido al tutorial', title='Bienvenido')
    sg.popup('En este tutorial aprenderás a jugar el juego', title='Tutorial')
    sg.popup('En este juego debes de colocar las minas en el tablero', title='Tutorial')
    sg.popup('Para colocar una mina debes de dar click en el boton de colocar minas', title='Tutorial')
    while True:
        event, values = window.read()
        if event == sg.WIN_CLOSED or event == 'Salir':…
        elif(event == 'Retroceder un movimiento'):…
        elif(event == 'Colocar minas'):…
        elif(event == 'Siguiente Paso'):…
```

La tienda hace un get hacia c++ y obtiene los datos de la tienda cargada en el json y crea una tabla con todos los objetos ordenados por categoría.

```
#Tienda
def tienda():      You, la semana pasada • Get Tienda
    sg.theme('DarkTeal4')
    res = requests.get(f'{base_url}ObtenerTienda/')
    data = res.text#convertimos la respuesta en dict
    data = json.loads(data)
    categoria = []
    combo = []
    for i in data:
        categoria.append([i,data[i]])
        combo.append(i)
    articulos = llenar_articulos(categoria[0])
    layout = [[sg.Text('TIENDA',size = (20,1), font="Arial 30 bold",justification='center')],…
    window = sg.Window('Tienda', layout, size=(600, 500), element_justification='center')
    while True:…

#Ver tutorial
```

Al elegir un objeto y seleccionar comprar se mostrara una ventana emergente con la imagen del objeto y una confirmación de la compra.

El menú de administrador muestra opciones diferentes a las del normal, desde aquí se podrán ver todos los reportes de usuarios y del tablero del tutorial.

```python
#Menu Admin
def menu_admin():      You, hace 2 semanas • API creada
    layout = [[sg.Text('Menu De Administrador',size = (20,0), font="Arial 30 bold", justification='center')],
              [sg.Button('Cargar Datos', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Iniciar Sesion', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Registrar Usuario', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Editar Usuario', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Eliminar Usuario', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Reportes', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Iniciar Juego', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Tienda', size = (20,0), font="Arial 15 bold")],
              [sg.Button('Salir', size = (20,0), font="Arial 15 bold")]]
    window = sg.Window('Menu', layout, size=(500, 550), element_justification='c')

    while True:…
    window.close()
    return event
```