Fecha de entrega: 27 de noviembre del 2016

Documentación Externa

Proyecto Torneos de fútbol

Estudiante: Danny Xie Li – Carnet 2016086098
Profesor: William Mata
INSTITUTO TECNÓLOGICO DE COSTA RICA
ESCUELA DE COMPUTACIÓN
CURSO TALLER DE PROGRAMACIÓN

Contenido

Enunciado del proyecto

A) Configuración del torneo

El usuario del programa va a dar los siguientes datos de configuración para un torneo específico:

- El nombre del torneo

- La cantidad de equipos participantes

- La cantidad de equipos que clasifican

- Puntos ganados por cada partido ganado

- Puntos ganados por cada partido empatado

- Lista de los equipos

- Planilla de Jugadores de cada equipo

Puede ser modificada solo la planilla de jugadores. La configuración puede ser consultada.

B) Generar el calendario de juegos

El programa creará automáticamente las fechas y los juegos de los partidos, según las condiciones definidas en el enunciado del proyecto entregado por el profesor.

C) Registrar los resultados de cada partido

Por cada partido jugado se registrará el resultado: código del equipo que es casa y goles anotados, código del equipo que es visita y goles anotados. También los nombres de los jugadores que anotaron los goles por cada equipo.

D) Tabla de posiciones

Se mostrará los resultados en esta tabla, los equipos que posean más puntos se ubicarán en el primer lugar.

E) Tabla de goleadores

Se mostrará la tabla de goleadores, según la cantidad de goles que anotaron.

F) Ayuda

Esta opción es usada para mostrar el manual de usuario.

G) Acerca de

Esta opción la usaremos para desplegar información "Acerca del programa".

H) Salir

Esta opción se puede usar para salir del programa.

Temas investigados

Interfaz Gráfica usada: Librería Tkinter de Python.

Elementos Usados:

- **Tk()** : Esta opción le permite al programador crear una ventana.
  ¿Cómo se usa?
  Cuando se desea crear una ventana llama está función y se lo asigna una variable (es recomendado) para futuros usos. Por ejemplo:
  Ventana = Tk ( )

- **Variables globales:** Es un tipo de variable libre en la cual se orienta en el ámbito del programa principal.
  ¿Cómo se usa?
  En el programa principal se define la variable global, que debe contener estos 3 elementos: el nombre de la variable, la asignación y el valor. Por ejemplo:
  Lista = [ ]
  Si se quiere usar esta variable en una función se llama esta variable por medio de la palabra global. Por ejemplo:
  global Lista

- **Toplevel (Ventana)**: Esta opción permite al programador crear una pantalla secundaria a la pantalla principal.
  ¿Cómo se usa?

Cuando se desea crear una pantalla secundaria al principal se escribe esa función y entre paréntesis la pantalla principal o la pantalla que desea relacionarlo. Por ejemplo:
VentanaSecundaria = Toplevel(Ventana)

- **Button (opciones)**: Esta opción le permite al programador crear botones y agregarlos a la pantalla, en donde se le puede agregar diferentes opciones como el tamaño de la letra, color del botón, la acción que desea realizar y en que pantalla se debe contener.

  ¿Cómo se usa?

  A la opción Button se le da las opciones entre paréntesis y separados por comas, en donde lo desea ubicar, en que pantalla, color, letra, entre otros, además se le debe empacar el botón para que aparezca en la pantalla de lo contrario no existe. Por ejemplo:
  Botón = Button(Ventana, text="Haga click").pack()

- **Label (opciones):** A igual que el botón, esta opción le permite crear etiquetas de textos o imágenes, posee las mismas opciones que el botón.

  ¿Cómo se usa?

  A la opción de Label se le da unos conjuntos de opciones separados por coma y entre paréntesis todas las opciones que se desea asignar. Además se le debe empacar la etiqueta para que aparezca en la pantalla de lo contrario no existe. Por ejemplo:
  Etiqueta = Label(Ventana, text ="Hola").pack()

- **Canvas (opciones) y Frame (opciones):** Estas dos opciones le permite al programador crear espacios dentro de una pantalla principal, en la opción de Canvas() le permite crear objetos, agregar botones, etiquetas entre otros en cambio el Frame todo lo mencionado anteriormente con excepción con crear objetos.

  ¿Cómo se usa?

  MiniEspacio = Canvas(Ventana, bg = "blue").pack()
  MiniEspacio = Frame(Ventana, text = "Hola", fg = "red").pack()
  En las opciones se deben poner la ventana en la que se quiere insertar y empacarlo o ubicarlo en la pantalla según pixeles o filas y columnas.

- **PhotoImage (file = NombreDelArchivo):** En esta opción le permite al programador agregar imágenes a la pantalla por medio de esta función.

  ¿Cómo se usa?

  La función PhotoImage, entre paréntesis se escribe la palabra file, el símbolo de asignar y el nombre del archivo en formato string. Por ejemplo:

  Imagen = PhotoImage (file ="hola.gif")

- **Entry (opciones):** Esta función le permite al programador agregar campos de textos a la interfaz gráfica. Además para tomar el valor que está en el campo de texto se usa una variable de tipo string o tipo entero.

  ¿Cómo se usa?

  Para crear un campo de texto se le pone las opciones entre paréntesis, la variable y empacarlo.

  Variable = StringVar () #Se define la Variable como un valor string.
  Espacio = Entry (Ventana, Variable).pack () #Se le asigna el campo de texto a una variable.
  Variable.get () #Toma el valor que tiene la variable.
  Variable.set ("Adiós") #Pone un valor a la variable.

- **ttk.ComboBox (opciones):** Esta función crea algo similar a una list box pero no lo es, es un tipo de botón que despliega una tira de valores que le permite al usuario seleccionar un valor de todos los que tiene en esa caja.

  ¿Cómo se usa?

  Se llama la función ttk.ComboBox y entre paréntesis las opciones que desea agregar, A demás para agregarle valores se le asigna una lista de valores a la variable values. Por ejemplo:

  Valor = StringVar () #Se le asigna el valor de string a la variable.
  Combo = ttk.ComboBox (Ventana, textvariable = valor, values = ["Casa", "Trabajo", "Otros"]).pack () #Se le asigna el combobox a la variable.
  Valor.get () #Para conocer el valor que se escogió.

- **messagebox.show Tipo (titulo,mensaje):** Esta función le permite al programador mandar un cuadro de texto dependiendo del tipo que se quiere mandar puede ser de error, información, advertencia, entre otros.

  ¿Cómo se usa?

  Se llama a la función escribiendo messagebox.show, el tipo de cuadro de texto que se quiere mandar y entre paréntesis el título del cuadro en formato string y el mensaje que se quiere transmitir también en formato string. Por ejemplo:

  messagebox.showerror ("Error", "Debe escribir un nombre")

- **Radiobutton (opciones):** En esta función le permite al programador crear botones en donde el usuario puede escoger entre las opciones que existe, su forma es de un círculo (botón) y el enunciado.

  ¿Cómo se usa?

  Se escribe la función radiobutton y entre paréntesis las opciones que desea que contenga y además se le debe poner un valor al radiobutton y una variable para poder conocer su valor. Por ejemplo:

  Radiobutton (ventana,text=texto, variable=IntString,value=1).pack()

- **os.system (ubicación, nombre del archivo):** Es un módulo que le permite al programador ejecutar archivos, en el programa de contactos fue usado para ejecutar sonidos o música en formato .wav.

  ¿Cómo se usa?

  Se debe llamar la función de os.system (), y en formato string la ubicación del archivo y el nombre del archivo. Por ejemplo:

  os.system ("start E:\BeethovenMoonlight.wav")

- **ttk.Notebook (opciones):** El propósito de este widget es permitir que el usuario pueda seleccionar ventanas de contenidos haciendo click en los diferentes pestañas.

  ¿Cómo se usa?

  Se llama el widget y entre paréntesis las opciones que desea agregarle.

  nbook = ttk.Notebook(ventana1).pack() #El widget se le asigna a la variable.

```
frame1 = Frame (ventana1) #Se debe crear una ventanilla

tab1=nbook.add (frame1, image = buscar,padding = 10)  #Se le
agrega una pestaña al widget .
```

- **Startfile("Nombre del Archivo"):** Esta opción le permite al programador abrir archivos.
  ¿Cómo se usa?

  Se escribe la función y entre paréntesis y en formato string el nombre del archivo que desea abrir. Por ejemplo:

```
startfile("Manual-De-Usuario-DContact.pdf")
```

- **Scrollbar(opciones):** Esta función le permite crear un scrollbar a la ventana, se le debe poner en las opciones la ventana en donde la desea poner y otras opciones si lo desea.
  ¿Cómo se usa?
  Se pone la función, la ventana en donde lo desea ubicar, en que orientación (vertical o horizontal). Por ejemplo:

```
  Barra1 = Scrollbar (frame2, orient = "vertical", command =
canvas1.yview)
```

- **ttk.Scale(opciones):** Esta opción le permite definir una escala de valores y debe poner el rango de valores que desea tener esto.
  ¿Cómo se usa?
  Se escribe la función, y le pone valores que desea, como el comando, los rangos de valores, la ventana, variable, tamaño, entre otras opciones.
  Ttk.Scale(ventana,from_=0, too_=100, length=200, variable=dificultad, command=DefinirDificultad).place(x=35,y=70)

- **Image.open(archivo):** Esta opción le permite al usuario abrir el imagen que desea poner, puede ser en formato .png, .jpg, .gif. Este crea un objeto.
  ¿Cómo se usa?
  Se escribe la función, y entre paréntesis se pone el nombre del archivo en formato string.
  Imagen.open("Hola.png")

- **ImageTk.PhotoImage(variable):** Esta opción le permite al usuario insertar una imagen, en donde crea la imagen según el nombre de la imagen asignado en la variable.
  ¿Cómo se usa?

Se escribe la función y entre parénesis el objeto asignado a la variable.
Imagen = Imagen.open("Hola.png")
Box = ImageTk.PhotoImage(Imagen)

- **SimpleDocTemplate (nombreArchivoPDF):** Esta opción le permite crear el archivo .pdf.
  ¿Cómo se usa?
  Se escribe la función y entre paréntesis se escribe el nombre del archivo con extensión .pdf en formato string. Por ejemplo:
  SimpleDocTemplate ("DMaster-Mind-Top-10-Detalles.pdf")

- **Table(datosEnLista):** Esta opcion le permite crear una tabla en el pdf, según la lista que pones.
  ¿Cómo se usa?
  Se escribe la función Table, y entre paréntesis la información en formato string. Por ejemplo:
  Table("String")

Librerías usadas

Antes de usar los elementos de la librería se deben importarlas, con la palabra import:

- ***import random***: Esta librería permite obtener datos aleatorios.

- ***Import sys o import os:*** Esta librería permite accede a funcionalidades del sistema operativo, que nos permite manipular la estructura de directorios. Import startfile: Permite abrir archivos.

- ***import tkinter.ttk as ttk:*** Esta librería contiene diferentes versiones de los elementos standares de Tkinter.

- ***from tkinter import *:*** Esta librería nos permite accede a todos los componentes graficas que posee Tkinter para desarrollar una interfaz gráfica.

- ***Import time***: Esta librería le permite acceder todos los contenidos relacionados con el tiempo, hora, fecha, entre otros elementos.

- ***From os import starfile***: Esta opción le permite acceder todos los contenidos de os, en donde especifica que le permite abrir archivos, en el caso de este trabajo archivos.pdf.

- ***from PIL import Image, ImageTk:*** Esta librería le permite al usuario acceder a formatos nuevos de imágenes, en el caso usado en el trabajo se usa para poder insertar imágenes en formato .png.

- ***from reportlab.lib import *:*** esta librería le permite al usuario manipular archivos .pdf, crear archivos pdf, modificar, eliminar, entre otras opciones.

- ***from tkinter import messagebox***: Esta libreria le permite al usuario usar cuadros de textos de informacion, error, advertencia, entre otros de la libreria Tkinter.

Librerías relacionadas con ***reportlab.lib,*** son usadas para importar opciones

- ***from reportlab.platypus import SimpleDocTemplate, Table, TableStyle, Paragraph***: Importa tablas, documentos pdf, parapoder crear tablas en un pdf.

- ***from reportlab.lib.styles import getSampleStyleSheet:*** Permite tener un estilo para crear el pdf, un formato de orden.

- ***from reportlab.pdfgen import canvas:*** Permite crear pantallas o permite dibujar, crear objectos.

Opciones de Labels, Buttons, Entrys, ComboBox, ventanas, entre otros.

.build(): Esta opción le permite crear el documento pdf. Por ejemplo documento.build(agregar), donde el documento será el que desea crear y agregar son los datos que se van a agregar.

.title (): Esta opción le permite ponerle un título a la ventana, debe ser en formato string y entre paréntesis. Por ejemplo: ventana.title ("Hola").

.geometry (): Está opción le permite definir el tamaÑo de la ventana ancho x altura en formato string. Por ejemplo: ventana.geometry ("500x100").

.maxsize (): Esta opción le permite definir el tamaño máximo que se puede expandir la ventana. Por ejemplo: ventana.maxsize (ancho,altura).

.iconbitmap (): Esta función le permite al programador agregarle un icono a la ventana, debe ser en formato string y el archivo en formato ico. Por ejemplo: ventana.iconbitmap ("Hola.ico").

Relief =: Esta opcion le permite agregarle relieve a los componentes de tkinter como botones, etiquetas, campos de texto, entre otros. El relieve se debe poner todo en mayúscula. Por ejemplo: relief=FLAT.

Width=: Está opcion le permite definir el tamaño del elemento en dimension a X. Por ejemplo: width = 20.

Height =: Está opcion le permite definir el tamaño del elemento en dimension a Y.Por ejemplo: height = 50.

Bg=: Esta opción le permite al programador definir el fondo de color de los componentes de Tkinter. Se debe poner el color en formato string y en minúscula. Por ejemplo: bg = "blue"

Fg=: Esta opción le permite al programador definir el color de letra ya sea en botones, campos de textos, etiquetas, etc. Por ejemplo: fg = "blue".

Font=: Esta opción le permite al programador definir el tipo de letra, familia de letra. Entre strings el tipo de letra y separado por coma el tamaño. Por ejemplo: Font = ("Helvetica",10)

Image=: Esta opción le permite al programador definir una imagen o agregar una imagen a un elemento, puede ser usados en botones, etiquetas, ventanas, etc. Se le asigna la variable que contiene la imagen la función image. Por ejemplo: Image = Salud.

Command=: Esta opción le permite al programador definir la función que desea realizar el botón. Se le debe asignar la palabra command a una función. Por ejemplo: command=saludar.

Row=: Esta opción le permite ubicar elemento en la posición de la fila que desea empezando desde 0 oeste a este. Esta función solo se usa con .grid(). Por ejemplo: grid (row=1,column=0).

Column=: Esta opción le permite ubicar elemento en la posición de la fila que desea empezando desde 0 norte a sur. Esta función solo se usa con .grid (). Por ejemplo: grid (row=1, column=0).

Text=: Esta opción le permite mostrar texto en un botón, etiqueta, etc. Debe asignarle a la variable text, el string que desea que se muestre. Por ejemplo: text = "Alo"

Textvariable=: Es un instancia de StringVar() que se asocia con un texto en un botón, etiqueta, entre otros. Si la variable cambia, el nuevo valor será mostrado en el elemento. Por ejemplo: textvariable = cambio. Para obtener el valor de la variable se usa .get() y para poner un valor a la variable .set().

Clases y archivos

Clase Jugador: esta clase es usada para validar los datos de entradas del jugador en donde son 3 datos; el nombre, la posición y el número.

Clase Equipo: Esta clase es usada para validar los equipos que se agregan a la variable global, son tres datos; el código del equipo, el nombre del equipo y la posición del equipo.

Clase torneo: la clase torneo, es una clase definida para validar los datos de entrada cuando se crea un torneo. Son cinco datos: el nombre del torneo, la cantidad de equipos participantes, la cantidad de equipos que clasifican, los puntos por partidos ganados y los puntos por partidos empatados.

Clase JugadorAnotador: Esta clase es usada para validar los datos de los jugadores que anotan en un partido, son 2 datos; el nombre del jugador y el país.

Clase ListaEsta: La siguiente clase es usada para verificar si una lista se encuentra en la variable global fechas o en la variable global ResultadosDePartidos.

Clase PosicionEscalon: Esta clase es usada para ordenar una lista de equipos según la posición en la escala, de menor a mayor.

Clase SonIguales: esta clase es definida para ordenar una lista de equipos según los puntos de diferencias, de mayor a menor.

Clase PuntosEquipos: Esta clase es usada para crear listas con los puntos de cada equipo, los puntos son los siguientes: juegos jugados, juegos perdidos, juegos empatados, juegos ganados, goles a favor, goles en contra, goles de diferencia y puntos.

Archivos

- **resultados_torneos_de_futbol.txt**

  Es usado para almacenar los resultados de un torneo, su formato es de la siguiente manera:
  [[“Código del equipo en casa”, Goles marcados del equipo en casa, “Código del equipo de visita”, Goles marcados del equipo de visita, [[“Jugador”, “Equipo que marcó”],[“Jugador”, “Equipo que marcó”],…[“Jugador”, “Equipo que marcó”]]]]

- **configuracion_torneos_de_futbol.txt**

  Este archivo de texto es usado para almacenar la configuración del torneo, su formato es de la siguiente manera:
  [“Nombre del torneo”, Cantidad de Equipos participantes, Cantidad de equipos que clasifican, Pts. por partidos ganados, Pts. por paridos empatados,[[“Código del equipo”,”Equipo”, Posición en la escala], [“Código del equipo”,”Equipo”, Posición en la escala], …, [“Código del equipo”,”Equipo”, Posición en la escala]]]

3. Conclusiones

Durante el desarrollo de este proyecto se encontró problemas en la creación de fechas en la sección del calendario, pero se solucionó al crear un notebook y una función que iba iterando pestañas al notebook, así se obtenía la cantidad de fechas que se ocupaba.

Además otro problema que se encontró fue el desarrollo del algoritmo de ordenamiento usando árboles, este no se pudo solucionar, así que no se usó árboles en la parte de tabla de goleadores. El problema fue de como representar los goles en un árbol y después como poder desplegarlos en orden para poder mostrarlo.

Entre los conocimientos que se adquirieron, fue desarrollar el proyecto usando el paradigma de programación en orientada objetos, se concluyó que el usó de este es más eficiente durante el desarrollo del código en comparación al desarrollo de funciones independientes y funciones auxiliares.

Además se amplió el conocimiento en las interfaces graficas usando la librería tkinter. También se aprendió un poco acerca de árboles binarios como estructura para organizar información.

4. Estadísticas de tiempo

Concepto	**Horas**
Configuración del torneo: datos generales	5
Configuración del torneo: información de equipos	6
Generar el calendario y desplegarlo	10
Registrar los resultados: marcadores	6
Registrar los resultados: goleadores	4
Tabla de posiciones	6
Tabla de goleadores (al menos para esta tabla se debe implementar con árboles binarios un algoritmo de ordenamiento, ponga los comentarios respectivos)	6
Ayuda (incluye manual de usuario)	3
Documentación del proyecto	3
TOTAL	53
Partes desarrolladas adicionalmente	4

5. Rúbrica de evaluación y análisis de resultados

Concepto	**Puntos**	**Puntos obtenidos**	**% Avance**	**Análisis de resultados**
Configuración del torneo: datos generales	5		100	
Configuración del torneo: información de equipos	15		100	
Generar el calendario y desplegarlo	15		100	
Registrar los resultados: marcadores	10		100	
Registrar los resultados: goleadores	10		100	
Tabla de posiciones	20		100	
Tabla de goleadores (al menos para esta tabla se debe implementar con árboles binarios un algoritmo de	15		70	No se pudo implementar con arboles binarios.
Ayuda (incluye manual de usuario)	5		100	
Documentación del proyecto	5		100	
TOTAL	**100**			
Partes desarrolladas adicionalmente				