# Carlos Lucio Junior ITS 220: Programming Languages & Concepts

# Prof: Mathew Loschiavo

**Final Exam**

Using the concepts that you have learned throughout this course and the chapters in your textbook, create a simple database search application that will return specified data of your choice. You will be given the freedom and the flexibility to choose what type of data that will be in your application.

Review Chapters 4 and 5 for database concepts and examples.

Review Chapters 5, 6, 7, 8, 8.5, and 9 for GUI-based concepts and examples.

The database structure of your application is like a spreadsheet with rows and columns, where each row is a record of entry, and each column is a specific field or attribute of your data. You are not required to create a relational database, which is much more complex and beyond the scope of this course.

You have several options that you can create on your database (not required to implement):

* 1. Develop a simple GUI-based interface for your application.
  2. You can use a text file to input your data.

You are to write 1-2 complete paragraphs, explaining in detail, on the type of data you have chosen and how your database will output that data. Some examples are, but are not limited to, a personal address book, a movie or book database, etc. You can use an example in the textbook as a template or model, but you cannot use that actual example for the Final.

You must submit both the source code (Link to code on Github) and perform at least two test cases with each of their respective results. Include your source code and screenshots of your test cases in a single document. Your test cases must validate that your program is working correctly. For full points, the source code and the test cases must be submitted.  Code organization, execution patterns, code complexity, data structures, and happy path/error handling will be factors for assessing/losing points. Partial credit will be given based on what you have submitted by midnight on Sunday, June 27, 2022. Best wishes and have fun!

.

**The Ride a Mountain Application**

The Ride a Mountain application was developed for the ones who have the desire to practice Snowboarding in the accessible Ski Resorts around California.  
It is an app to suggest the best mountains based on your skill levels.  
The skill levels will be determined around questions regarding the sport in the format of a quiz. Each question will give you a certain score and that is how the program will determine what your “Ridding” skill levels is.  
The levels are divided in 5 different categories; NOOB, BEGINNER, INTERMEDIATE, ADVANCED and EXPERT.  
Depending on your category, the program will provide you the best mountain recommended to you extracting the mountains information from a specific database.

Each mountain information also contains the website that it’s being captured from a different column in the database file.  
With the information the user can choose between exit the program or start over.

**The database file and command lines.**

# Importing session from and excel spreadsheet  
import pandas as pd  
  
from tkinter import \*  
from tkinter import messagebox  
  
#Variables definitions for questions. Defining values to determine the results and to bring the correct information from the table.  
def result():  
 result= int(gear.get())+int(goofy.get())+int(practice.get())+int(groomer.get())  
 name = str(depot.get())  
 mtrecommend\_df = pd.read\_excel(r"C:\Finalexame\finaldata.xlsx", engine='openpyxl')

Table

Description automatically generated with medium confidence

**Ride a Mountain Program logic – Brainstorm – My very 1st Draft!**

**The idea of the program is to recommend the right mountain around California according to the skills levels of the Boarders.  
  
It is a 4-question quiz about Snowboarding that will determine the skill level between: Beginner/Noob, Intermediary, Advanced and Expert.  
  
Knowledge Questions = kq1, kq2, k3 and k4  
1. Do you need to rent your Snowboard gear?  
2. Do you know what Regular and Goofy means?  
3. Have you ever practiced Skateboard or Surf before?  
4. Do you know what is a groomer?  
  
Results = Score  
if kq1 = no = 10  
if kq2 = yes = 10   
if kq3 = yes = 10  
if kq4 = yes = 10  
  
total = kq1+kq2+kq3+kq4  
  
if score = 40 result = table 1 from Excel or Txt file  
if score = 30 result = table 2 from Excel or Txt file  
if score = 20 result = table 3 from Excel or Txt file  
if score = 10 result or 0 = table 4 from Excel Txt file**

**The Code with my comments – 113 lines.**#################################################################  
# File name:Carlos-Final-Exame-SDUIS.py #  
# APP Name = Ride a Mountain #  
# Author: Carlos Lucio Junior #  
# Date:06-22-2021 #  
# Classes: ITS 220: Programming Languages & Concepts #  
# Instructor: Matthew Loschiavo #  
# This is a Final Exame #  
# Create a game to teach kindergartners how to sum #  
# You can find all the files used in this code at #  
# https://github.com/J4rn3s/FinalExame #  
#################################################################  
# Importing session from and excel spreadsheet  
import pandas as pd  
  
from tkinter import \*  
from tkinter import messagebox  
  
#Variables definitions for questions. Defining values to determine the results and to bring the correct information from the table.  
def result():  
  
 valid = True  
  
 while valid == True:  
 try:  
 result= int(gear.get())+int(goofy.get())+int(practice.get())+int(groomer.get())  
 valid = False  
 except ValueError:  
 messagebox.showwarning('Alert','Looks like you are missing something. Make sure you answered all the questions') # Validation if it is not a letter  
 break  
  
 name = str(depot.get())  
 mtrecommend\_df = pd.read\_excel(r"C:\Finalexame\finaldata.xlsx", engine='openpyxl')  
  
 if result == 0:  
 messagebox.showinfo('Level Result', "Noob")  
 Output.insert(END, "No problem, "+name+"\nEveryone has to start from somewhere. The recommended mountain for you is Snow Valey. ")  
 a = (mtrecommend\_df.loc[mtrecommend\_df['score'] == result, ['description','site']])  
 b = str(a.to\_string(index=False, header=False))  
 Output.insert(END, b)  
  
 elif result == 10:  
 messagebox.showinfo('Level Result', "Beginner")  
 Output.insert(END, "That is great "+name+"\n,looks like you know something. The recommended Mountain for you is Snow Valley. ")  
 c = (mtrecommend\_df.loc[mtrecommend\_df['score'] == result, ['description', 'site']])  
 d = str(c.to\_string(index=False, header=False))  
 Output.insert(END, d)  
  
 elif result == 20:  
 messagebox.showinfo('Level Result', "Intermediate")  
 Output.insert(END, "Nice "+name+"\n, looks like you have some skills. The recommended Mountain for you is Mountain High. ")  
 e = (mtrecommend\_df.loc[mtrecommend\_df['score'] == result, ['description', 'site']])  
 f = str(e.to\_string(index=False, header=False))  
 Output.insert(END, f)  
  
 elif result == 30:  
 messagebox.showinfo('Level Result', "Advanced")  
 Output.insert(END, "Sick "+name+".\n We can tell that you at least ride 4 fun. The recommended Mountain for you is Big Bear / Snow Summit. ")  
 g = (mtrecommend\_df.loc[mtrecommend\_df['score'] == result, ['description', 'site']])  
 h = str(g.to\_string(index=False, header=False))  
 Output.insert(END, h)  
  
 elif result == 40:  
 messagebox.showinfo('Level Result', "Expert")  
 Output.insert(END,"Dope "+name+"!\n You are probably preparing yourself for the X-Games! The recommended Mountain for you is Mammoth! ")  
 i = (mtrecommend\_df.loc[mtrecommend\_df['score'] == result, ['description', 'site']])  
 j = str(i.to\_string(index=False, header=False))  
 Output.insert(END, j)  
  
#Exit Application Button  
def ExitApplication():  
 MsgBox = messagebox.askquestion('Exit Application', 'Are you sure you want to exit the application',icon='warning')  
  
 if MsgBox == 'yes':  
 app.destroy()  
 else:  
 messagebox.showinfo('Return', 'You will now return to the application screen')  
  
#def sos(): #Here I tried to implement a save button for the results. It did not work.#  
 ## messagebox.showinfo('SOS', 'CALL 911')  
  
app = Tk()  
app.title('Ride a Mountain')  
Label(app, text = "Enter your name",bg="light blue").pack()  
depot = Entry(app)  
depot.pack()  
  
#Question 1  
Label(app, text = "Do you need to rent your Snowboard gear?").pack()  
gear = StringVar()  
gear.set(None)  
Radiobutton(app, variable = gear, text = "Yes", value = 0).pack()  
Radiobutton(app, variable = gear, text = "No", value = 10).pack()  
  
#Question 2  
Label(app, text = "Do you know what Regular and Goofy means?").pack()  
goofy = StringVar()  
goofy.set(None)  
Radiobutton(app, variable = goofy, text = "Yes", value = 10).pack()  
Radiobutton(app, variable = goofy, text = "No", value = 0).pack()  
  
#Question 3  
Label(app, text = "Have you ever practice Skateboard or Surf before?").pack()  
practice = StringVar()  
practice.set(None)  
Radiobutton(app, variable = practice, text = "Yes", value = 10).pack()  
Radiobutton(app, variable = practice, text = "No", value = 0).pack()  
  
#Question4  
Label(app, text = "Do you know what is a groomer?").pack()  
groomer = StringVar()  
groomer.set(None)  
Radiobutton(app, variable = groomer, text = "Yes", value = 10).pack()  
Radiobutton(app, variable = groomer, text = "No", value = 0).pack()  
  
Output = Text(app, height=15,width=68,bg="light gray")  
Output.pack()  
  
Button(app, text = "Results", command = result, bg="blue", fg="white").pack()  
#Exit Application Window  
Button(app, text='Exit Application', command=ExitApplication, bg='brown', fg='white').pack()  
#Button(app, text = "SOS", command = sos).pack() #Part of the save button#  
app.mainloop()

Test results.  
1. Happy path  
Graphical user interface, application

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, application

Description automatically generated

Text

Description automatically generated with medium confidence

Missing answer test. error handling example.

except ValueError:  
 messagebox.showwarning('Alert','Looks like you are missing something. Make sure you answered all the questions') # Validation if it is not a letter  
 break

Graphical user interface, application

Description automatically generated

Exit Application button tests  
Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

All the results were performed successfully.  
A copy of all the files project is inside my GITHUB repository.  
Please follow the link below for your appreciation.

<https://github.com/J4rn3s/FinalExame>

If you would like to have access to the whole content I have been working/studying on, please follow the link below.

<https://github.com/J4rn3s?tab=repositories>