

Forum:[Hack Forums](#)

Thread: [Help me write this Python program please](#)

Pkgoingout - July 9, 2019, 3:25 pm

I got an Assignment for one of you guys if you can help me
Send me a message on Skype @ live:MichaelRazz1

Use Class Objects to store information for each student
Store all student objects in a list `student_list`
read student information from the input file of exam 2
(at least 4 students, each with at least 4 courses and 4 acct items)

ask user to enter a command (and student ID if needed)
commands are as followed
I - display student personal info
C - display courses taken
G - display letter grades for courses taken and GPAA
B - account balance

Command with no ID required
T - total # of students
A - account balance of all students (add balance of all students)
Q - quit program

Filename as followed if you want to be graded (word document only)
`Your_name_exam_2.docx`

Need Flowchart (screen shot/image accepted) and Pseudo-code
Please copy and paste code and results (no screen shot for this part)
Test all commands

Hint

using the code, practice creating student objects with different info, print
append students to `student_list`
`student_list=[]`
`student_list.append(student_a)`
For the exam,
1) write code to read and print student data,
2) copy code from (1) add some code to create student as you read, append
3) write an infinite loop for user command input, print the command, "quit" to exit
4) add code to (3) to implement the command, implement any commands that you can do

#Courses

Code class students:

```
def __init__(self, idn): # constructor - must be the first method
    self.sid = idn # creates student object with idn
    self.courses = [] # init
```

```

self.acct = []
self.fn = ""
self.ln = ""
self.addr = ""

def display_info(self): #accessor method
    print(self.sid)

def get_sid(self): #accessor method
    return self.sid

def set_personal(self, first, last, address): #mutator method
    self.fn = first
    self.ln = last
    self.addr = address

def get_personal(self): #accessor method
    info = []
    info.append(self.fn)
    info.append(self.ln)
    info.append(self.addr)
    return info

def set_course(self, course_info): #mutator method
    self.courses.append(course_info)

def get_courses(self):
    return self.courses

def get_average(self,cid):
    for i in range(len(self.courses)):
        if cid in self.courses[i]:
            return cal_avg(self.courses[i]) #found
    return -999 #not found

def cal_avg(score):
    total = 0
    score_list = score.split()
    count = len(score_list) - 2 # scores start at 3rd element
    for i in range(2, len(score_list)): # tart from 2 to end
        total += int(score_list[i])
    #    print(total,int(score_list[i]))

    return int(round(total / count, 0)) # round average

def main():
    student_a = students("12345")

```

```

student_a.display_info()
asid = student_a.get_sid()
print("sid ", asid)
student_a.set_personal("John","Hung","200 Elm St., Davie, FL 33066")
a_info=student_a.get_personal()
print(a_info)
student_a.set_course("COP1006 201903 77 86 92 75 88 99")
student_a.set_course("ENG1006 201902 92 76 75 86 82 75 88 99")
c_info = student_a.get_courses()
print(c_info)
avg = student_a.get_average("COP1006 201903")
print("COP1006 201903 Average score ",avg)
avg = student_a.get_average("ENG1006 201902")
print("ENG1006 201902 Average score ",avg)
avg = student_a.get_average("MAT2006 201902")
print("MAT2006 201902 Average score ", avg)

```

```
main()
```

```

12345
sid 12345
['John', 'Hung', '200 Elm St., Davie, FL 33066']
['COP1006 201903 77 86 92 75 88 99', 'ENG1006 201902 92 76 75 86 82 75 88 99']
COP1006 201903 Average score 86
ENG1006 201902 Average score 84
MAT2006 201902 Average score -999

```

#students example

Code class students:

```

def __init__(self, idn): # constructor - must be the first method
    self.sid = idn # creates student object with idn
    self.courses = [] # init
    self.acct = []
    self.fn = ""
    self.ln = ""
    self.addr = ""

```

```

def display_info(self): #accessor method
    print(self.sid)

```

```

def get_sid(self): #accessor method
    return self.sid

```

```

def set_personal(self, first, last, address): #mutator method
    self.fn = first
    self.ln = last
    self.addr = address

```

```

def get_personal(self): #accessor method
    info = []
    info.append(self.fn)
    info.append(self.ln)
    info.append(self.addr)
    return info

def set_course(self, course_info): #mutator method
    self.courses.append(course_info)

def get_courses(self):
    return self.courses

def get_average(self,cid):
    for i in range(len(self.courses)):
        if cid in self.courses[i]:
            return cal_avg(self.courses[i]) #found
    return -999 #not found

def cal_avg(score):
    total = 0
    score_list = score.split()
    count = len(score_list) - 2 # scores start at 3rd element
    for i in range(2, len(score_list)): # tart from 2 to end
        total += int(score_list[i])
    #    print(total,int(score_list[i]))

    return int(round(total / count, 0)) # round average

def main():
    student_a = students("12345")
    student_a.display_info()
    asid = student_a.get_sid()
    print("sid ", asid)
    student_a.set_personal("John","Hung","200 Elm St., Davie, FL 33066")
    a_info=student_a.get_personal()
    print(a_info)
    student_a.set_course("COP1006 201903 77 86 92 75 88 99")
    student_a.set_course("ENG1006 201902 92 76 75 86 82 75 88 99")
    c_info = student_a.get_courses()
    print(c_info)
    avg = student_a.get_average("COP1006 201903")
    print("COP1006 201903 Average score ",avg)
    avg = student_a.get_average("ENG1006 201902")
    print("ENG1006 201902 Average score ",avg)
    avg = student_a.get_average("MAT2006 201902")
    print("MAT2006 201902 Average score ", avg)

main()

```

12345

sid 12345

['John', 'Hung', '200 Elm St., Davie, FL 33066']

['COP1006 201903 77 86 92 75 88 99', 'ENG1006 201902 92 76 75 86 82 75 88 99']

COP1006 201903 Average score 86

ENG1006 201902 Average score 84

MAT2006 201902 Average score -999