21 Final Review

May 7, 2019

1 CS 21 Final Review

2 # Question 1

Write a complete program that plays a guessing game with numbers. The goal is for the user to guess a randomly selected number between 1 and 100 as quickly as possible. The program will inform the user whether the hidden number is higher or lower than the current guess. You must break the problem up into a main() function and at least one additional function.

```
In [1]: import random

        def main():
            accum = 0
            computer = random.choice(range(1, 101)) #remember the end is an open interval
            guessed = False

            while guessed == False:
                user_s = input("What's your guess? ")
                accum += 1
                if check(user_s) == True:
                    guessed = eval(int(user_s), computer)
                else:
                    while True:
                        user_s = input("Invalid, try again: ")
                        accum += 1
                        if check(user_s) == True:
                            guessed = eval(int(user_s), computer)
                            break

            print("You guessed the answer in %d tries!" %(accum))


        def eval(number, computer):
            """ Evaluates the number to see if it is correct or not. """
            if number == computer:
                print("Correct!")
                return True
            elif number < computer:
                print("The number is higher than %d" %(number))
                return False
            else:
                print("The number is lower than %d" %(number))
                return False

        def check(user_s):
            """ User input validation. """
            if user_s.isdigit() == True:
                user = int(user_s)
                if  0 <= user <= 100:
                    return True
            return False



        main()
```

```
What's your guess? pony
Invalid, try again: -1
Invalid, try again: cheto
Invalid, try again: 101
Invalid, try again: 300
Invalid, try again: 33
The number is higher than 33
What's your guess? 50
The number is higher than 50
What's your guess? 74
The number is higher than 74
What's your guess? 90
The number is lower than 90
What's your guess? 80
The number is lower than 80
What's your guess? 75
Correct!
You guessed the answer in 11 tries!
```

3 # Question 2

Student Class should include: id, lastname, credits, course load.

```
In [2]: class Student(object):
            """ A class to represent student course selections. """

            def __init__(self, ids, name):
                """ Make a new student given their name and year. """
```

```
        self.id = ids
        self.lName = name
        self.credits = 0
        self.courseLoad = 0

    def registerCurrent(self, number):
        """ Sets the current course load. """
        if number > 0 and self.courseLoad <= 4:
            self.courseLoad = number

    def withdraw(self):
        """ Decreases course load by 1. """
        if self.courseLoad > 0:
            self.courseLoad -= 1

    def passedCourse(self):
        """ Removes one course from the current course load and
        adds to the student's overall course-credits. """
        if self.courseLoad > 0:
            self.courseLoad -= 1
            self.credits += 1

    def createEmail(self):
        """ Creates an email for the student. """
        email = "" # accounting for pyhton not being strongly typed
        email = self.lName + str(self.id) +"@swarthmore.edu"
        return email

    def __str__(self):
        """ Return a string representing the student. """
        s = "ID: %d \n Name: %s \n Course Load: %d \n Earned Credits: %d" %(self.id, se
                                                                            self.cours
        return s

def main():
    gake = Student(1, "Janes")
    gake.registerCurrent(4)
    gake.withdraw()
    gake.passedCourse()
    email = gake.createEmail()
    print(gake)
    print("This student's email is %s" %(email))

    tasty = Student(12, "cheto")
    tasty.registerCurrent(4)
    tasty.withdraw()
    tasty.passedCourse()
    tasty.passedCourse()
```

```
            tasty.passedCourse()
            tasty.registerCurrent(4)
            tasty.withdraw()
            email = tasty.createEmail()
            print(tasty)
            print("This student's email is %s" %(email))


        if __name__ == '__main__':
            main()
```

```
ID: 1
 Name: Janes
 Course Load: 2
 Earned Credits: 1
This student's email is Janes1@swarthmore.edu
ID: 12
 Name: cheto
 Course Load: 3
 Earned Credits: 3
This student's email is cheto12@swarthmore.edu
```

4 # Question 3

Write recursive and iterative versions of a function that, given a string and a character, returns a new string with the character before and after each letter in the string.

```
In [3]: def main():
            word = input("Please input the word: ")
            char = input("Please input the char: ")
            recAns = recursive(word, char)
            iterAns = iterative(word, char)
            print("recursive answer: %s \n iterative answer: %s" %(recAns, iterAns))


        def recursive(word, char):
          if len(word) == 0:
            return word
          elif len(word) == 1:
            return char+word+char
          else:
            return char+word[0] + recursive(word[1:], char)

        def iterative(word, char):
            if len(word) == 0:
                return ""
            retStr = char
            for i in word:
                retStr += i + char
            return retStr


        if __name__ == '__main__':
            main()
```

```
Please input the word: hello
Please input the char: *
recursive answer: *h*e*l*l*o*
 iterative answer: *h*e*l*l*o*
```

Jake's Note: Coming up with the recursive answer is a bit too hard for the CS21 Final. To quote one of the 21 professors "If I were giving this on an exam, I would probably make the output just h*ello**, and skip the first char." Here is what you would be excpetced to write.

```
In [ ]: def recursive(word, char): #only spits out h*e*l*l*o when given hello and *
            if len(word) == 0:
                return ""
            else:
                s = recursive(word[1:], char)
                ret = char+word[0]+s
                return ret
```

5 # Question 4

Write a function to take a word-count list and return the word that has the highest count.

```
In [3]: def main():
            wc = [['frank', 99], ['ruby', 300], ['doug', 37], ['clayton', 297], ['owo', 100],
            word = highestCount(wc)
            print(word)

        def highestCount(lst):
            """returns the highest value in a list of lists"""
            maxVal = 0
            word = 0
            for subLst in lst: #remember that lists are iterable objects
                if subLst[1] > maxVal:
                    maxVal = subLst[1]
                    word = subLst[0]
            return word

        if __name__ == '__main__':
            main()


ruby
```