CPSC 1301, Computer Science I Programming Exercise

Week 03, Recursive Methods

This activity consists of four programming exercises. The following exercises are open book and open note. You are free to use any written documentation you wish. However, these are individual exercises, and you cannot consult with each other in writing your programs. Name your program `RecurMethods_lastname.py`

This programming exercise has four parts consisting of four requirements. The grade for each requirement is indicated, for a maximum of 100 points. At a minimum, your program must compile successfully and run.

A *recursive* function is a function that calls itself. Recursive activities are extremely common in every day life. When you mow your lawn, you cut one mower-width and see if you are finished. If so, you stop and drink a beer. If not, you cut one mower-width and see if you are finished, again and again. When you shop for groceries, you place an item in your cart and mark it off the list. If it's the last item, you check out. If not, you place an item in your cart and mark it off the list, again and again. Recursive functions work exactly the same way. You call the function, perform one task, and check to see if you are done. If so, you stop. If not, you call the function again. I have completed the first part of this exercise below, and you can see how simple this is.

Warm up Create a console application that will accept ten numbers between 0 and 100, and report their sum. The program below illustrates one way in which this can be done. Create this program and make sure you understand how it works.

```
1  def sumit(count, sum):
2      if count > 10:
3          return sum
4      else:
5          print("Enter number", count)
6          number = int(input())
7          return sumit(count + 1, sum + number)
8
9  if __name__ == '__main__':
10     sum = 0
11     count = 1
12     print("The sum of ten numbers is", sumit(count, sum))
```

Requirements These are the requirements of this exercise:

- Write a function that accepts a float and returns a character, input of 90.333 will output 'A' and input of 79.9 will output a 'C'.
- Write a function that takes three parameters (start, end, accum) as input, sums ten integers, and returns the sum of the ten integers.
- Modify this function so that the end parameter is user defined, sums integers entered by the user, and returns the sum of the integers.
- Write a function that accepts two parameters (counter, accum) as input, sums integers until some stop value, and returns the sum of all integers entered before the stop value.

Starter template You may use the following as your starter template:

```
1  #!python
2  # Name: RecurMethods.py
3  # Author: <your name>
```

```
4  # Date: <date>
5  # Purpose: does grading by recursive methods
6
7  def hello():
8      pass
9
10 def sum_ten(start, end, accum):
11     pass
12
13 def sum_any(counter, accum):
14     pass
15
16 def letter_grade(num_grade):
17     pass
18
19 #main function executes the defined functions
20 if __name__ == '__main__':
21     hello()
22     print()
23
24     print("calling sum_any() and calculating the final letter grade")
25     start = pass
26     end = pass
27     accum = pass
28     total_grades = sum_any(start, end, accum)
29     average_grade = total_grades / end
30     let_grade = letter_grade(average_grade)
31     print("Total of grades is", total_grades)
32     print("Average of grades is", average_grade)
33     print("Letter grade is", let_grade)
34     print()
35
36     print("calculating the final letter grade for a user defined limit")
37     start = pass
38     print("How many grades do you have: ")
39     end = int(input())
40     accum = pass
41     total_grades = sum_ten(start, end, accum)
42     average_grade = total_grades / end
43     let_grade = letter_grade(average_grade)
44     print("Total of grades is", total_grades)
45     print("Average of grades is", average_grade)
46     print("Letter grade is", let_grade)
47     print()
48
49     print("calculating the final grade for an arbitrary number of grades, no pre-specified
           limit")
50     accum = pass
51     counter = pass
52     (counter, total_grades) = sum_any(counter, accum)
53     average_grade = total_grades / counter
54     let_grade = letter_grade(average_grade)
55     print("Total of grades is", total_grades)
56     print("Average of grades is", average_grade)
57     print("Letter grade is", let_grade)
58     print()
```

Sample output Your sample output should match the following:

```
Hello from 'RecurMethods.py'


calling sum_ten() and calculating the final letter grade
1 Enter the next grade of 10 grades:
88
```

```
2 Enter the next grade of 10 grades:
99
3 Enter the next grade of 10 grades:
89
4 Enter the next grade of 10 grades:
87
5 Enter the next grade of 10 grades:
50
6 Enter the next grade of 10 grades:
99
7 Enter the next grade of 10 grades:
98
8 Enter the next grade of 10 grades:
97
9 Enter the next grade of 10 grades:
96
10 Enter the next grade of 10 grades:
95
Total of grades is 898
Average of grades is 89.8
Letter grade is B

calculating the final letter grade for a user defined limit
How many grades do you have:
5
1 Enter the next grade of 5 grades:
78
2 Enter the next grade of 5 grades:
79
3 Enter the next grade of 5 grades:
80
4 Enter the next grade of 5 grades:
90
5 Enter the next grade of 5 grades:
99
Total of grades is 426
Average of grades is 85.2
Letter grade is B

calculating the final grade for an arbitrary number of grades, no pre-specified limit
Enter the next grade, -1 to exit:
55
Enter the next grade, -1 to exit:
99
Enter the next grade, -1 to exit:
88
Enter the next grade, -1 to exit:
77
Enter the next grade, -1 to exit:
100
Enter the next grade, -1 to exit:
100
Enter the next grade, -1 to exit:
-1
```

```
Total of grades is 519
Average of grades is 86.5
Letter grade is B
C:\Users\ccc31\cols-st\cpsc1301\python-progs>
```