SI@UCF Introduction to Programming in Python

Test #2 Solution
July 21, 2014

1) (15 pts) What is the output of the following segment of code?

```
str = "ABCDEFGHIJ"
print(str[2:7])
print(str[:8])
print(str[3:])
print(str[-6:-2])
print(str[:-5])
```

CDEFG
ABCDEFGH
DEFGHIJ
EFGH
ABCDE (Grading: 3 pts each, 2 pts if off by 1, 1 pt if off on both ends)

2) (15 pts) What is the output of the following segment of code?

```
a = 1
b = 2
c = a + b
for i in range(5):
    print("a =",a,"b =",b,"c =",c)
    a = b
    b = c
    c = a + b
```

a = 1 b = 2 c = 3
a = 2 b = 3 c = 5
a = 3 b = 5 c = 8
a = 5 b = 8 c = 13
a = 8 b = 13 c = 21

Grading: 1 pt per number, no exceptions.

3) (15 pts) Write a segment of code that determines (and prints out) the smallest value of n such that the sum 1 + 1/2 + 1/3 + 1/4 + 1/5 + … + 1/n > 6.

```
total = 0                           # 2 pts
term = 0                            # 1 pt
while total <= 6:                   # 3 pts
    term = term + 1                 # 3 pts
    total = total + 1/term          # 4 pts
print(term)                         # 2 pts
```

4) (15 pts) Write a segment of code that prompts the user to enter a secret number. Continue doing so until the guess they make is exactly 20 higher than the last guess they made. At this point, tell them that they've won. (Thus, one must play at least 2 turns to win.)

```
prevnum = int(input("Enter your first guess.\n"))
nextnum = int(input("Enter your next guess.\n"))
while nextnum - prevnum != 20:
    prevnum = nextnum
    nextnum = int(input("Enter your next guess.\n"))
print("You win!")
```

Grading: 3 pts for each line except the last.

5) (15 pts) A perfect number is a number whose sum of proper divisors is equal to the number itself. For example, 6 is a perfect number since all of the numbers that divide into it that are less than it add to 6: 1 + 2 + 3 = 6. A second example of a perfect number is 28, since 1 + 2 + 4 + 7 + 14 = 28. Write a program that prints out all of the perfect numbers less than 10000. (Note: On the off hand chance that you have these memorized, please don't just write print statements to print these out. You won't get credit for doing so. Please solve this the intended way, with loops that check each number from 1 to 10000 for the desired property.)

```
def main():

    for num in range(1,10001):

        total = 0
        for i in range(1,num):
            if num%i == 0:
                total += i
        if total == num:
            print(num)

main()
```

6) (9 pts) Roughly how many times is the code segment below going to print "Go Knights!"? Give some proof for your answer.

```
n = 1000000
while n > 0:
    print("Go Knights!")
    n = n//2
```

This runs roughly 20 times. Each time, n is getting divided by 2, until it reaches 0. This is the same as starting at 1 and multiplying by 2 until reaching a million. Since 2^{20} is slightly more than a million, we can ascertain that this runs either 20 or 21 times.

Grading: 5 pts for being in the right ballpark, 4 pts for the reasoning

7) (15 pts) A simple formula for the height of an object dropped from Y feet high t seconds after its dropped (on Earth) is $f(t) = Y - 16t^2$. Fill in the program below so that it prints out a chart of the height of an object after each second, starting with t = 0. The user will enter the initial height. The last line should print a height of 0, regardless of what the mathematical equation says, since the object won't fall below the ground.

```
def main():

    height = int(input("Enter the initial height.\n"))
    t = 0;

    saveheight = height                    # 3 pts

    print("Time\tHeight")
    while height > 0:

        print(t,"\t",height)
        t = t + 1                          # 4 pts
        height = saveheight - 16*t*t       # 5 pts


     print(t,"\t",0)                       # 3 pts
```

```
main()
```

8) (1 pt) In what shape are the individual pieces of the popular cereal Froot Loops?

Loops