Computer Science 125
10 September 2020

1. **How is a for loop different from a while loop?**

For loop: iterates fixed number of times

while loop: iterate while some condition is true

2. **Simulate rolling a fair 6-sided die until a 5 appears. Print each roll and the total.**

```
roll = 0  # initializes variable roll
total = 0
while 5 has not appeared:        roll != 5
    roll die
    print roll
    total = total + roll
print total
```

```
import random
roll = 0
total = 0
while roll != 5:
    roll = random.randrange(1,7)
    print("roll is",roll)
    total += roll
print("total is",total)
```

3. **DANGER: Infinite loops!**

```
total = 0
while total != 5:
    total = 6
```

```
while True:
    print("Hi")
```

4. **How would you convert the following for loop to a while loop?**

```
for i in range(1, 10, 2):
    print(i*i)
```

result:
1
9
25
49
81

```
i = 1
while i < 10:
    print(i*i)
    i = i + 2
```

5. **Iterating over characters in a string:**

```
for c in "Hello":
    print c
```

OUTPUT:
H
e
l
l
o

PRACTICE WITH LOOPS – SOLUTIONS

CS 125

Working with a partner/group, use the following steps to solve each of the following problems.

(a) Plan your function on the white board (either on the classroom wall or on Zoom). Write out your entire program. Think about what errors might occur and how to fix them.

(b) Plan multiple test cases for your function. What input will you send to your function? What value should the function return?

(c) *Only after you have completed steps (a) and (b) should you type your code in Python.*

(d) After you have typed your function, run your test cases. Does your function work? If not, how can you fix it?

1. Write a function `computeSum(n)` that accepts a number n as a parameter and uses a loop to figure out how many terms in the sum $1 + 2 + 3 + \cdots$ are necessary for the sum to exceed n. Your function should return the number of terms. For example:

   `computeSum(6)` returns `4`

   `computeSum(17)` returns `6`

```
def computeSum(n):
    i=0
    sum = 0

    while sum <= n:
        i = i + 1
        sum = sum + i
    return i

print("It takes", computeSum(6), "terms to exceed 6.")
print("It takes", computeSum(17), "terms to exceed 17.")
```

2. Write a program that prompts the user for an integer between 1 and 10. However, if the user enters something that is *not* an integer between 1 and 10, your program should print an error message and prompt the user to try again. Your program should repeat this as many times as necessary, until the user enters an integer between 1 and 10.

```
n = int(input("Enter an integer between 1 and 10: "))

while not(n >= 1 and n <= 10):
    print("Please try again.")
    n = int(input("Enter an integer between 1 and 10."))

print("Thank you.")
```

3. Write a program that asks the user for a string of lowercase letters and uses the `isVowel()` function from the previous class to determine whether the string consists entirely of vowels.

```
def isVowel(ch):
    if ch == "a" or ch == "e" or ch == "i" or ch == "o" or ch == "u":
        return True
    return False

def onlyVowels(text):
    for c in text:
        if not isVowel(c):
            return False
    return True

text = input("Enter a string of lowercase letters:")
if onlyVowels(text):
    print("Your string consists entirely of vowels.")
else:
    print("Your string does not consist entirely of vowels.")
```

4. Use a loop to investigate the following sum:

$$1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} + \cdots$$

What happens when you add up many terms of this sum? How many terms are necessary to obtain a sum greater than 1.99? How many terms are necessary to exceed 1.9999? Do you think the sum will ever exceed 2?

```
def computeSum(limit):
    i=0
    sum = 0

    while sum < limit:
        sum = sum + 1/(2**i)
        i = i + 1
        # print(i, sum)   #testing

    return i

lim = float(input("Enter a limit:"))
terms = computeSum(lim)
print("It takes", terms, "terms to exceed", lim)
```