CS 1200 FS18 HW 1

Due 2018-09-04 (Tuesday) at 11:59 PM

Submit your assignment to Canvas:

1. A PDF file that contains all the answers to the individual questions, all pictures, all code, and all code output. This should all be well-organized. Points will be deducted for sloppy or disorganized work.
2. All the Python codes (.py file) (You may put all codes in one .py file).

If you need a program that helps you put PDF files together into a single PDF file, try http://www.pdfsam.org/. The program there is open source and available for free.

1. (25 points) Simple questions about Python.

   (a) (6 points)Trace the output of the following program:

   ```
   word = "foobar"
   print  word[-2]
   print  word[2]
   print  word[0:2]
   print  word[:2]
   print  word[::2]
   print  word[::-1]
   ```

   (b) (4 points) What is printed by the following function if it is given the number 1 as n?

   ```
   def numLegs(n):
       if n == 0:
           print("haha you can not walk!")
       if n == 1:
           print( "you hop!")
       if n == 2:
           print("that is normal")
       else:
           print("you must trip yourself")
   ```

   >>> numLegs(1)

(c) (5 points) Trace the output of the following program:

```
for s in ["b", "c"]:
    for n in [1, 4]:
        print s*n,
```

(d) (4 points) Trace the output of the following program:

```
a = [1, 2]
b = []
b.append(a)
b.append(a)
a.append(3)
print b
```

(e) (6 points) Use a for loop to print the decimal representations of $1/2, 1/3, \ldots, 1/10$, one on each line.

2. (15 points) Fermat's Last Theorem says that there are no integers a, b, and c such that

$$a^n + b^n = c^n$$

for any values of n greater than 2.

(a) Write a function named fermatCheck that takes four parameters a, b, c, and n and that checks to see if Fermat's theorem holds. If n is greater than 2 and it turn out to be true that

$$a^n + b^n = c^n$$

the program should print something like " Fermat was wrong!" Otherwise the program should print something like "That doesn't work".

(b) In *The Simpsons* episode *The Wizard of Evergreen Terrace* Homer writes the equation $3987^{12} + 4365^{12} = 4472^{12}$ on a blackboard, which appears to be a counterexample to Fermat's Last Theorem. Use fermatCheck in (a) to check whether $3987^{12} + 4365^{12} = 4472^{12}$ is true.

3. (15 points)Write a Python function called altDif that produces the alternating difference of a list of numbers. For example,altDif([7 6 9 10])=7-(6-(9-10))=7-(6-(-1))=7-7=0; altDif([3 5 4])= 3 - (5 - 4) = 3 - 1 = 2; altDif([6 7]) = 6-7 = -1.

4. (15 points) Consider the following recursive function:

```
def f(x):
    if x > 100:
        return x - 9
    else:
        return f(f(x+10))
```

   (a) Determine the following values by running the above code: f(0), f(-9), f(45), f(99), f(100), and f(250).

   (b) Give a simpler non-recursive definition of f that gives the result. Without giving a formal proof at this point, try to give some reason why you think that the two definitions produce the same result.

5. (15 points) Using only the most primitive Python functions write a purely recursive function called SuperReverse. Its input should be an arbitrary list. Its output should be a list which is the reverse of the original list, plus every list found anywhere in the original list must be reversed. For example, suppose that the starting list is [[1,9], [5, [6, 2]], 3] the output list should be [3, [[2, 6], 5], [9,1]]. You may not use any looping control structures such as for or while loops. You must use only the most basic and primitive Python commands.

6. (15 points) The Ackermann function provides a good test of the recursive capabilities of your system. The Ackermann functin is written in Python called Q as shown below:

```
def Q(x,y):
    if x == 0:
        return y+1
    if y == 0:
        return Q(x-1,1)
    else:
        return Q(x-1,Q(x,y-1))
```

   (a) Test your system by trying to generate the values of Q(X,Y) as X and Y both range from 0 to 5. What is Q(5,5)? Find the place where your system begins to take an inordinate amount of time. Watch for the computer hanging up.

   (b) At first glance it might not be obvious that Q(X,Y) always halts for all values of X and Y. To better understand how the Q function works, calculate Q(2,2) by hand and list the number of calls to Q that it makes. (Note: the proof that Q(X,Y) always halts is given later in this course.)