Assignment #6

General Instructions. For this assignment, you will be writing 3 separate programs. They should be submitted as 3 different files using NYU Classes. The files should be named:

Lastname-Firstname-assign6-1.py
Lastname-Firstname-assign6-2.py
Lastname-Firstname-assign6-3.py

1. Vending Machine (6 points).

Your program should first prompt the user to enter the price of an item in a vending machine. Then the program should repeatedly prompt the user to deposit coins (nickels, dimes, and quarters) until the total is equal to or more than the price. Before each prompt, the program should print out how much money the user still needs to deposit. If the input is invalid, your program should print “Invalid amount”, and continue to accept input.

At the end, the program should print out any change owed to the user.

Here is sample input and output:

Enter item price: 1.39

Amount left: 1.39
Please deposit a nickel, a dime, or a quarter (n, d, q) : q
Amount left: 1.14
Please deposit a nickel, a dime, or a quarter (n, d, q) : q
Amount left: .89
Please deposit a nickel, a dime, or a quarter (n, d, q) : q
Amount left: .64
Please deposit a nickel, a dime, or a quarter (n, d, q) : q
Amount left: .39
Please deposit a nickel, a dime, or a quarter (n, d, q) : q
Amount left: .14
Please deposit a nickel, a dime, or a quarter (n, d, q) : d
Amount left: .04
Please deposit a nickel, a dime, or a quarter (n, d, q) : n

Your change is .01
2. Estimate the value of $\pi$ (7 points).

The value of $\pi$ is:

$$3.141592653589793238462643383279502...$$

One way of estimating the value of $\pi$ is to calculate a finite number of terms of an infinite sum. For example, you can use the Leibniz formula and its implementation below to estimate $\pi$:

```python
# $\pi = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} ...$

terms = int(input("Enter the number of terms: "))

sign = +1
pi_approx = 0
for denom in range(1,terms*2+1, 2):
    pi_approx += sign * 4/denom
    sign = -sign
print("pi = ", pi_approx)
```

Here is a formula for estimating the value of $\pi$ that uses an infinite product instead of an infinite sum:

$$\pi/2 = \frac{2}{1}*(\frac{2}{3})*(\frac{4}{3})*(\frac{4}{5})*(\frac{6}{5})*(\frac{6}{7})*(\frac{8}{7})*(\frac{8}{9}) ...$$

Write a program to estimate $\pi$ using this formula. Your program should first prompt the user to enter a number of terms to use in the estimate. Then all of these terms should be successively calculated in a loop and multiplied to produce a running product. To get the final estimate of $\pi$, your program should multiply the final result by 2, and then print out this final estimate.

You should test your program on different numbers of terms (e.g. 100, 1000, 10000, etc) You will know if your program is correct if the result gets more accurate (closer to $\pi$) as the number of terms increases.

**Hint:** One approach is to calculate two terms on each iteration. For example, on the first iteration, calculate $(2/1) * (2/3)$, on the second iteration, calculate $(4/3) * (4/5)$, etc.
3. **Calendar pretty-printer (7 points).**

Write a program to print out a calendar month, for a month with 31 days.

Your program first prompts the user to enter a number (1-7), indicating which day of the week the month starts on. Your program should verify that the number is in range, and if it isn’t, prompt the user until a valid number is entered. Then your program should print out the month. **Your program must use nested loops to get full credit (either while or for loops).**

The numbers in the calendar should line up, that is, they should be right-justified. **Hint:** Use the `format()` function to line up each number.

**Examples:** If the month starts on day 3, then the output should look like:

```
  1  2  3  4  5
  6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```

If the month starts on day 7, then the output should look like:

```
  1
  2  3  4  5  6  7  8
  9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31
```

If the month starts on day 1, then the output should look like:

```
  1  2  3  4  5  6  7
  8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```