Python HW #4

Name: Christopher Welch

1. Given the two lists:

listOne = [3, 6, 9, 12, 15, 18, 21]
listTwo = [4, 8, 12, 16, 20, 24, 28]

Create a third list called listThree by adding all odd-index elements from the first list and all even index elements from second list. Print out your answer.

Your code should print out:

The third list is: [6, 12, 18, 4, 12, 20, 28]

```
listOne = [3, 6, 9, 12, 15, 18, 21]
listTwo = [4, 8, 12, 16, 20, 24, 28]
indexer = 0
listThree = []
for x in listOne:
    if x % 2 == 0:
        listThree.append(x)
for y in listTwo:
    if indexer == 0 or indexer % 2 == 0:
        listThree.append(y)
    indexer += 1
print("The third list is:" , listThree)
```

```
The third list is: [6, 12, 18, 4, 12, 20, 28]
Press any key to continue . . .
```

2. Given the following dictionary:

speed ={'jan':47, 'feb':52, 'march':47, 'April':44, 'May':52, 'June':53, 'july':54, 'Aug':44, 'Sept':54}

Get all values from the dictionary and add it to a new list called speedList, but don't add duplicates values to the new list.

Your code should print out:

speedList [47, 52, 44, 53, 54]

```python
speed ={'jan':47, 'feb':52, 'march':47, 'April':44, 'May':52, 'June':53,
'july':54, 'Aug':44, 'Sept':54}
speedList = []
for item in speed.values():
    if item not in speedList:
        speedList.append(item)
print("speedList", speedList)
```

3. Given the following list of numbers:

numberList = [10, 20, 33, 46, 55, 61, 70, 88, 95]

First iterate the list and print only those numbers, which are divisible of 5. Then iterate the list and print only those numbers that are odd.

Your code should print out:

```
Numbers from numberList that are only divisible of 5 are:
10
20
55
70
95
Numbers from numList that are odd are:
33
55
61
95
```

```python
numberList = [10, 20, 33, 46, 55, 61, 70, 88, 95]
print("Numbers from numberList that are only divisible of 5 are:")
for x in numberList:
    if x % 5 == 0:
        print(x)
print("Numbers from numList that are odd are:")
for x in numberList:
    if x % 2 != 0:
        print(x)
```

4. For the given list:

item_list= [2, 6, 13, 17, 22, 25, 29, 31, 38, 42, 47, 55, 59, 63, 68, 71, 77, 84, 86, 90, 99]

Write python code to do a binary search over the given list, where the search value is input by a user (positive integers from 1 to 100 only). Print out the results if the search found the number or not.

```
def getUserVal1to100():
    inputVal= int(input("Enter a Search value: (positive integers from 1 to 100 
only)"))
    while inputVal < 1 or inputVal > 100 or inputVal != int(inputVal):
        inputVal= int(input("You Entered an Invalid Value.\nEnter a search value: 
(positive integers from 1 to 100 only)"))
    return inputVal

def binarySearch(iVal, valList):
    start = 0
    end = len(valList)
    found = False
    while start <= end and not found:
        mid = (start + end) // 2
        if valList[mid] == iVal:
            found = True
            return print(iVal, "was found at index: ", mid)
        elif iVal < valList[mid]:
            end = mid - 1
        else:
            start = mid + 1
    if found == False:
        print(iVal, "was not found")
    return found

item_list= [ 2, 6, 13, 17, 22, 25, 29, 31, 38, 42, 47, 55, 59, 63, 68, 71, 77, 84, 
86, 90, 99]
inputVal = getUserVal1to100()
binarySearch(inputVal, item_list)
```

5. The national debt is a hot topic this year, both because it continues to set new records and because this is the beginning of the 2020 presidential election season. Just how big is the debt? It is a very big number, and its size makes it difficult to comprehend. Let's write some python code that helps us understand this big number.

According to http://www.usdebtclock.org, the US national debt is ~$22 trillion dollars. For this exercise, we will use the $100 bill (The thickness of every U.S. denomination bill is 0.0043 inches) to determine the height in miles of a stack $100 bills required to equal the debt number of $2.7 trillion dollars. Write python code that determines this answer and print your answer out with the appropriate label. Additionally, compare your answer to the average distance of the moon from Earth (238,857 miles) as a multiple of your answer and print this out tool, with the appropriate label.

```python
#The Debt is $22,632,047,000,000 or ~$22.6 Trillion
nationalDebt = 22632047000000       # $22,632,047,000,000
a100BillThicknessInches = 0.0043    # in Inches
inchesPerMile = 63360
milesOf100sNeeded = nationalDebt / 100 * a100BillThicknessInches / inchesPerMile
print("The height of a stack $100 bills required to equal the National Debt of 
${:,.2f} is:".format(nationalDebt))
print("\t\t\t {:,.2f}".format(milesOf100sNeeded), "Miles")
#Additionally, compare your answer to the average distance of the moon from 
Earth (238,857 miles)
distanceOfMoonFromEarth = 238857    # 238,857 miles
print("The amount of times we can make it from the Moon to the Earth with that 
many miles is:")
print("\t\t\t {:,.2f}".format(milesOf100sNeeded / distanceOfMoonFromEarth), 
"Times")
```