```
def getArea(length,width):
    area = length * width
    return area
print(getArea(10,5))
a = getArea (100,50)
v = a*10
print (v)
```

```
50
50000
```

```
'''
Example of function
'''

def countCoolWarm(data):
    countCool = 0
    countWarm = 0
    for temperature in data:
        if temperature <=15:
            countCool += 1
        elif temperature >= 26:
            countWarm += 1
    return countCool, countWarm

## Test
sample = [35,40,25,37]
coolDays, warmDays = countCoolWarm(sample)
print(coolDays, warmDays)
```

```
0 3
```

```
'''
A function that calls
'''
def getNumbers(numbers):
    tempValue = numbers.split()
    result = []
    for value in tempValue:
        result.append(float(value))
    return result

inputValue = input("Enter numbers separated by,")
numbers = getNumbers(inputValue)
print(numbers)

n2 = getNumbers("123 456 678 233 6567 232")
print(n2)
```

```
Enter numbers separated by, 1 2 3 4
[1.0, 2.0, 3.0, 4.0]
[123.0, 456.0, 678.0, 233.0, 6567.0, 232.0]
```

```
def saveAnswerKey(correctAns):
    pass

def getQuizAnswer():
    # Just to get things going without reading the file
    return ['a', 'b', 'b', 'c', 'a', 'b', 'b', 'c', 'd', 'd' ]

def showMenu():
    print("1. show")
    print("2. change")
    print("3. take test")
    print("0. Quit")

def showAnswer(correctAnswer):
    for answer in correctAnswer:
        print(answer)

def changeAnswerKey(changedAnswer):
    index = int(input("Enter index of answer: "))
    newAns = input("Enter new answer: ")
    changedAnswer[index - 1] = newAns

def takeTest(correctAns):
    # Ask user to enter 10 letters, save in a list
    # User a loop to compare the list with corrections, copy paste lab4qn2 done
    pass

## main
answerKey = getQuizAnswer()
option = -1
while option != 0:
    showMenu()
    option = int(input("Enter an option: "))
    if option == 1:
        showAnswer(answerKey)
    elif option == 2:
        changeAnswerKey(answerKey)
    elif option == 3:
        takeTest(answerKey)

saveAnswerKey(answerKey)
print("End of Program")



'''
A simple game that uses nested lists with multiple players
'''
import random
numPlayers = 4
maxPoint = 50
# [["p1", 20] , {"p2", 30], {..}, ...]
result = []
for idx in range(numPlayers):
    name = "player{}".format(idx + 1)
    result.append([name, 0])

# Start the game now
```

```
totalPoint = 0
currPlayer = 0 # Index of the current player
while totalPoint < 50:
    point = random.randint(1, 6)  # Roll a dice
    totalPoint += point
    player = result[currPlayer]
    player[1] += point
    print(player[0], point, player[1], totalPoint)
    currPlayer += 1                 # Move to the next player
    if currPlayer == len(result):   # Go pass last player
        currPlayer = 0              # Back to first player

for p in result:
    print(p)



'''
Rewriting previous example using functions
'''
import random

def getGameData():
    numPlayers = int(input("How many players? "))
    maxPoint   = int(input("Max point per game? "))
    players    = []
    # Create players and put them in the list: players
    for idx in range(numPlayers):
        name = input("Name of player {}?".format(idx + 1))
        players.append([name, 0])

    return maxPoint, players

def updateScore(playerData, currPlayer, point, total):  # total correspond to
totalPoint
    player  = playerData[currPlayer]                    # Get the record of
current player
    player[1] += point                                  # Update his score
    print("{} current roll {} Total {}".format(player[0], point, player[1]))
    print("Game total is {}".format(total))

###### main starts here #####
maxPoint, playerData = getGameData()    # Call a function to get the inputs

# Get ready to start the game
totalPoint = 0
currPlayer = 0
# Start the game loop now
while totalPoint < maxPoint:
    point = random.randint(1,6)     # Roll a dice
    totalPoint += point
    updateScore(playerData, currPlayer, point, totalPoint)  # Add score to player
    currPlayer += 1
    if currPlayer == len(playerData):
        currPlayer = 0

# Game over, print the result
for p in playerData:
    print(p)
```

```
'''
Example: converting one set of values to another
'''
# Simpleest solution is to use if..elif..elif...

result = "OXOXO"
totalPoint = 0
for value in result:
    if value == "O":
        totalPoint += 5
    elif value == "X":
        totalPoint += 10




'''
Another exercise on functions and nested lists
This program allows user to enter a combination of 6 numbers
The program will check the combination against past winning combinations (fake
data, generated)
The program will report how many time the user (the combination) get 2 or more
numbers correct.
We will break the program into 2 functions: (you can break it down further if you
like)
    one function to generate past winning combinations
    one function to compare user's input against the past combinations
'''
import random

def getPastData(years):
    numbers = tuple(range(1,50)) # numbers = (1, 2, 3, ... 49)
    pastWinningNums = []    # To store all the winning combinations
    for idx in range(52 * years):  # Number of combinations to be generated
        weeklyResult = random.sample(numbers, 6)  # Randomly picks 6 numbers
        pastWinningNums.append(weeklyResult)

    return pastWinningNums

def getNumbers(inputValue):
#    listOfNumber = []
#    for item in inputValue.split():
#        listOfNumbers.append(int(item))

    listOfNumbers = []
    listOfString = inputValue.split()  # It becomes ["12", "2", "34"...]
    for item in listOfString:
        value = int(item)
        listOfNumbers.append(value)
    return listOfNumbers

def compare(userCombo, pastCombo):
    twoOrMoreMatches = []                       # Store matching combinations
    for combo in pastCombo:                     # Check each past combination
        matchCount = 0                          # Count how many numbers match
        for idx in range(len(userCombo)):       # Need a loop to compare numbers
in list
```

```
            if userCombo[idx] == combo[idx]:    # Value and position must be the
same
                matchCount += 1

            if matchCount >= 2:
                twoOrMoreMatches.append(combo)

    return twoOrMoreMatches

###### main starts here ######

# Get input and set up simulation data
years     = int(input("How many years of past data to generate? "))
pastCombo = getPastData(years)

# Start simulation now
inputValue = input("Enter 6 numbers: ")
userCombo = getNumbers(inputValue)
result = compare(userCombo, pastCombo)

'''Explanation:
User will enter (example): 12 34 43 1 8 3
But the input is stored as string "12 34 43 1 8 3"
We need to convert into list of numbers: [12, 34, 43, 1, 8, 3]
We need a function (compare) to check user's input against all the past
combination
The function will return a list of ombinations that match the user's
input(partially)
Therefore, we have result = compare(...)
In this case, result wil be a list of lists
'''
#Show simulation result
print(userCombo)
for combo in result:
    print(combo)



How many years of past data to generate? 20
Enter 6 numbers:    10 20 23 24 25 40
[10, 20, 23, 24, 25, 40]
[10, 20, 23, 24, 25, 40]
[48, 15, 23, 36, 24, 40]
[15, 20, 23, 22, 11, 8]
[15, 20, 23, 22, 11, 8]
[15, 20, 23, 22, 11, 8]
[15, 20, 23, 22, 11, 8]
[19, 39, 23, 45, 49, 40]
[37, 2, 23, 28, 25, 5]
[37, 2, 23, 28, 25, 5]
[33, 48, 30, 24, 25, 44]
[33, 48, 30, 24, 25, 44]
```