2014 Pre-release Homework

1. Task 1:

```
def GetChoiceFromUser():
  Choice = input('Do you think the next card will be higher than the last card (enter y or n)? ')
  return Choice.lower()[0]
```

2. Task 2:

```
def GetMenuChoice():
  Choice = input()
  print()
  return Choice.lower()[0]
```

3. Task 3a:

Question 1: The `def GetPlayerName():` function.

Question 2: Use a while loop and a boolean, and only accept the correct data type (string). As long as the input isn't accepted, the boolean will remain false and the while loop will keep asking for a suitable name.

Question 3: check_string, boolean data type.

Pseudocode:

```
def GetPlayerName() DO
    check_string <- False
    repeat
        PlayerName <- Input("Please enter your name: ")
        IF len(PlayerName) > 0 DO
            Output("Thank you")
            check_string <- True
            return PlayerName
        ELSE DO
            Output("That is invalid, please enter your name: ")
```

Program Code:

```
def GetPlayerName():
  check_string = False
  while check_string == False:
    PlayerName = str(input('Please enter your name: '))
    if len(PlayerName) > 0:
      print("Thank you")
      check_string = True
      return PlayerName
    else:
      ("That is invalid, please enter your name: ")
```

Task 3b:

Question 1: The `UpdateRecentScores` function.

Program Code:

```
def GetPlayerName():
  check = input("Would you like to enter your name? (y or n): ")
  if check.lower()[0] == "y":
    check_string = False
    while check_string == False:
      PlayerName = str(input('Please enter your name: '))
      if len(PlayerName) > 0:
        print("Thank you")
        check_string = True
        return PlayerName
      else:
        ("That is invalid, please enter your name: ")
  elif check.lower()[0] == "n":
    print("Thank you for playing.")
```

Task 4:

Program Code:

```
def DisplayRecentScores(RecentScores):
  print()
  print('Recent Scores: ')
  print()
  print("Name:          Score:          ")
  for Count in range(1, NO_OF_RECENT_SCORES + 1):
    print("{0:<14} {1:<15}".format(RecentScores[Count].Name, RecentScores[Count].Score))
  print()
  print('Press the Enter key to return to the main menu')
  input()
  print()
```

Task 5:

Question 1: `from datetime import date`

Question 2:

```
-ResetRecentScores(RecentScores):

-UpdateRecentScores(RecentScores, Score):

-DisplayRecentScores(RecentScores):

-GetPlayerName():
```

Question 3: Firstly you have to import the date module by using `from datetime import date`. Once you have a variable for the date e.g `date_of_birth`, you create another variable for the date type e.g `actual_date` and assign it to the following:

```
actual_date = datetime.strptime(date_of_birth, "%d/%m/%Y")
```

Program Code:

```
GetPlayerName()::

def GetPlayerName():
  check = input("Would you like to enter your name? (y or n): ")
  if check.lower()[0] == "y":
    check_string = False
    while check_string == False:
      PlayerName = str(input('Please enter your name: '))
      if len(PlayerName) > 0:
        print("Thank you")
        check_string = True
        date = datetime.now()
        return PlayerName, date
      else:
        ("That is invalid, please enter your name: ")
  elif check.lower()[0] == "n":
    print("Thank you for playing.")

ResetRecentScores(RecentScores)::

def ResetRecentScores(RecentScores):
  for Count in range(1, NO_OF_RECENT_SCORES + 1):
    RecentScores[Count].Name = ''
    RecentScores[Count].Score = 0
    RecentScores[Count].date = None

UpdateRecentScores(RecentScores, Score)::

def UpdateRecentScores(RecentScores, Score):
  PlayerName, date = GetPlayerName()
  FoundSpace = False
  Count = 1
  while (not FoundSpace) and (Count <= NO_OF_RECENT_SCORES):
    if RecentScores[Count].Name == '':
      FoundSpace = True
    else:
      Count = Count + 1
  if not FoundSpace:
    for Count in range(1, NO_OF_RECENT_SCORES):
      RecentScores[Count].Name = RecentScores[Count + 1].Name
      RecentScores[Count].Score = RecentScores[Count + 1].Score
      RecentScores[Count].date = RecentScores[Count + 1].date
    Count = NO_OF_RECENT_SCORES
  RecentScores[Count].Name = PlayerName
  RecentScores[Count].Score = Score
```

```
  RecentScores[Count].date = date

DisplayRecentScores(RecentScores)::

def DisplayRecentScores(RecentScores):
  print()
  print('Recent Scores: ')
  print()
  print("Name:          Score:          Date played:")
  for Count in range(1, NO_OF_RECENT_SCORES + 1):
    print("{0:<14} {1:<15} {2}".format(RecentScores[Count].Name, RecentScores[Count].Score, RecentScores[Count].date))
  print()
  print('Press the Enter key to return to the main menu')
  input()
  print()
```

Additional Task - Variable Roles:

Question 1:

Variable Role	Description	Example
Fixed value	A variable with a value which cannot be changed once it has been initialised.	`NoOfSwaps = 1000`
Stepper	A variable which is updated with each systematic succession of values.	`Score`
Most recent holder	A variable holding the most recent value encountered when processing a succession of unpredicatable values.	`Choice = "q"`
Most wanted holder	A variable holding a value which is most appropriate for solving a problem or continuing a program.	`Choice`
Gatherer	A variable that accumulates the effect of individual values.	`Count = Count + 1`
Transformation	A variable which gets its value from a fixed calculation of values from other variables.	`LastCard.Rank = NextCard.Rank`
Follower	A variable which receives its value from a previous variable which was updated and passed down the outdated value.	`RecentScores[Count + 1].Name`
Temporary	A variable which only holds a value for a short time until it is no longer needed.	`Deck[Position2].Rank`

Additional Task - Functions and Parameters:

Question 1:

The difference between passing by value and passing by reference is that when a variable is passed by reference it uses a reference to the same memory location as the parameter, and any changes are accessible from where the function call was made, whereas when passed by value it copies the value of the calling code's variable to the functions parameter. Changes made to the copy have no effect on the original variable.

Question 2:

Function parameter	Mechanism
`def GetRank(RankNo):`	Value
`def GetSuit(SuitNo):`	Value
`def LoadDeck(Deck):`	Reference
`def ShuffleDeck(Deck):`	Reference
`def DisplayCard(ThisCard):`	Reference
`def GetCard(ThisCard):`	Reference
`def GetCard(Deck):`	Reference
`def GetCard(NoOfCardsTurnedOver):`	Value
`def IsNextCardHigher(LastCard):`	Value
`def IsNextCardHigher(NextCard):`	Value
`def DisplayEndOfGameMessage(Score):`	Value
`def DisplayCorrectGuessMessage(Score):`	Value
`def ResetRecentScores(RecentScores):`	Reference
`def DisplayRecentScores(RecentScores):`	Reference
`def UpdateRecentScores(RecentScores):`	Reference
`def UpdateRecentScores(Score):`	Value
`def PlayGame(Deck):`	Reference
`def PlayGame(RecentScores):`	Reference