[SOLUTION] CS-465-AI-Lab-Midterm-CS A-Version[B]

March 9, 2023

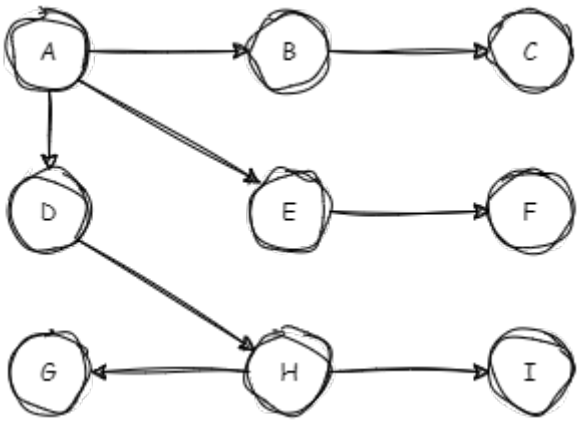
Question 1: Write a function that takes a temperature in Celsius as input and returns the equivalent temperature in Fahrenheit. The formula for converting Celsius to Fahrenheit is: F = (C * 9/5) + 32

[1]:
```
def celsius_to_fahrenheit(celsius):
    fahrenheit = (celsius * 9/5) + 32
    return fahrenheit

print(celsius_to_fahrenheit(37))
```

```
98.6
```

Question 2: Write a code of Breadth First Search (BFS) Algorithm. Test your code by traversing the following graph.



[2]:
```
graph = {
    "A" : ["B", "D", "E"],
    "B" : ["C"],
    "C" : [],
    "D" : ["H"],
    "E" : ["F"],
    "F" : [],
    "G" : [],
    "H" : ["G", "I"],
```

```
    "I" : []
}
```

[3]:
```
visited = [] # List to keep track of visited nodes.
queue = []     #Initialize a queue

def bfs(visited, graph, node):
    visited.append(node)
    queue.append(node)

    while queue:
        s = queue.pop(0)
        print (s, end = " ")

        for neighbour in graph[s]:
            if neighbour not in visited:
                visited.append(neighbour)
                queue.append(neighbour)

# Driver Code
bfs(visited, graph, 'A')
```

```
A B D E C H F G I
```

Question 3: You have data of car's sales in excel file as **"car__sales.csv"**. You are requested to show the data of only people above 30 years old and they have sold more than 5 cars.

[4]:
```
# importing the csv module
import csv

filename = "car_sales.csv"

# initialize Columns
fields = []

# initialize Rows
rows = []

# reading csv file
with open(filename, "r") as csv_file:

#   creating a csv reader object
    csv_reader = csv.reader(csv_file)

#   extracting field names through first row
    fields = next(csv_reader)
```

```
#   extracting each data row one by one
    for row in csv_reader:
        if int(row[2]) > 30 and int(row[6]) > 5:
            rows.append(row)

# printing the field names
print("\t".join(field for field in fields))

for row in rows:
#   parsing each column of a row
    for col in row:
        print(col + "\t", end = '')
    print()
```

```
ï»¿Sales ID     Name    Age     Gender  Years of sales experience       Number
of hours worked per day  Number of cars sold     Total value of cars sold
Total number of extras sold (insurance, parts, options, etc.)   Total value of
extras sold
3       Ricky   34      M       23      4       6       84000   2       230
23      Gabriel 64      M       40      8       8       115750  1       245
```

Question 4: Implement a class in Python to represent a point in 2D space. Write initializer with default value(s). The class should have accessor and mutators for the point, and method to calculate the distance between this point and another point.

•Run all methods to check your code.
•Create three instances of above class.
•Add above instances to new list.
•Iterate the above list.

[5]:
```
import math

class Point:
    def __init__(self, x, y):
        self.__x = x
        self.__y = y

    def set_coordinates(self, x, y):
        self.__x = x
        self.__y = y

    def get_coordinates(self):
        return self.__x, self.__y

    def distance_to(self, other_point):
        dx = self.__x - other_point.__x
        dy = self.__y - other_point.__y
        return round(math.sqrt(dx*dx + dy*dy), 2)
```

```
    def __str__(self):
        return f"Point [coordinates: {self.get_coordinates()}, distance to␣
 ↪origin: {self.distance_to(Point(0, 0))}]"
```

[6]:
```
point1 = Point(2, 3)              # create a new point with coordinates (2, 3)
point2 = Point(5, 7)              # create another point with coordinates (5, 7)
print(point1.get_coordinates())  # get the coordinates of the first point␣
 ↪(should be (2, 3))
print(point2.get_coordinates())  # get the coordinates of the second point␣
 ↪(should be (5, 7))
print(point1.distance_to(point2)) # calculate the distance between the two␣
 ↪points (should be approximately 5.0)

obj1 = Point(1, 8)
obj2 = Point(19, 10)
obj3 = Point(0, 6)
```

```
(2, 3)
(5, 7)
5.0
```

[7]:
```
my_list = [obj1, obj2, obj3]

for i in my_list:
    print(i)
```

```
Point [coordinates: (1, 8), distance to origin: 8.06]
Point [coordinates: (19, 10), distance to origin: 21.47]
Point [coordinates: (0, 6), distance to origin: 6.0]
```