INTERNSHIP TASKS

Day 11:

EXERCISE – 11

Name	: S. Deva Manikanta
Clg Id	: 12119003
Course	: Python
Org	: IGIAT – VSKP
Date	: 12-04-2024

Exercise Level 1

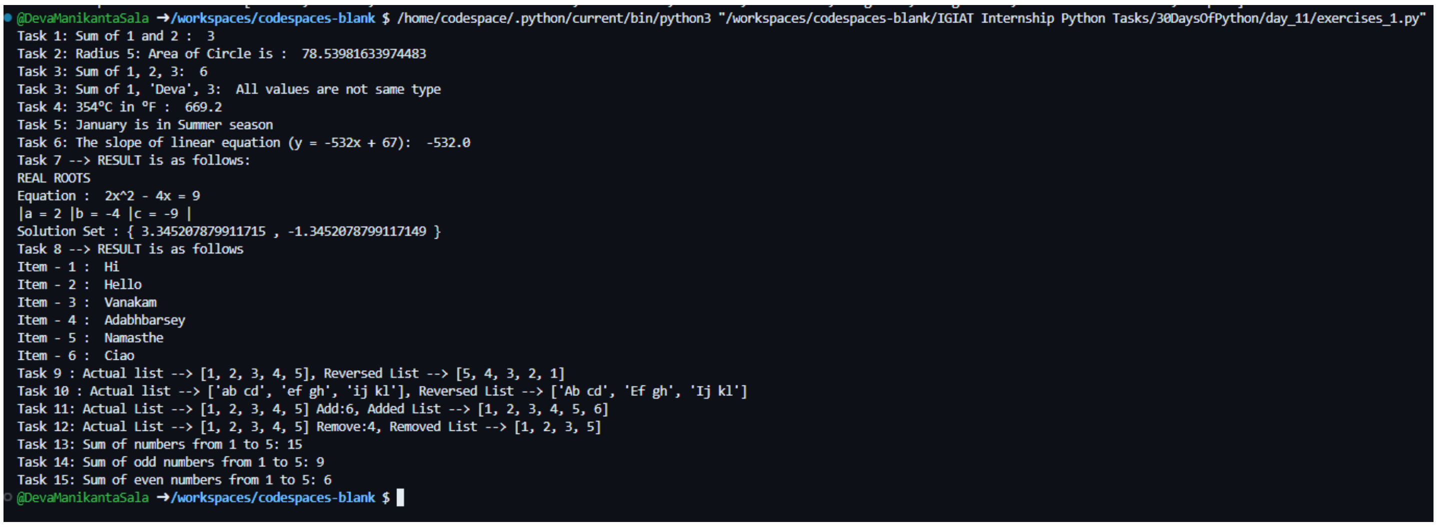
```
#Task 1:
#Declare a function add_two_numbers. It should take two parameters and it returns a sum
def add_two_numbers(a, b):
    return (int(a)+int(b));
print("Task 1: Sum of 1 and 2 : ", add_two_numbers(1, 2));

#Task 2:
#Declare the area of circle is calculated as follows: area = π * (r^2). Write a function that
calculates area_of_circle
import math as m;
def area_of_circle(radius):
    return (m.pi*(float(radius)**2));
print("Task 2: Radius 5: Area of Circle is : ", area_of_circle(5.0));

#Task 3:
#Write a function called add_all_nums which takes arbitrary number of arguments and sums all
the arguments. Check if all the list items are number. If not do give a reasonable feeback.
def add_all_nums(*numbers):
    all_are_same_type = all(isinstance(number, int) for number in numbers);
    if(all_are_same_type):
        return sum(numbers);
    else:
        return "All values are not same type";

print("Task 3: Sum of 1, 2, 3: ", add_all_nums(1, 2, 3));
print("Task 3: Sum of 1, \'Deva\', 3: ", add_all_nums(1, 'Deva', 3));

#Task 4:
#Write a function that converts °C to °F like convert_celsius_to_fahrenheit
def convert_celcius_to_fahrenheit(degrees):
    return float(degrees * (9/5) + 32);

print("Task 4: 354°C in °F : ", convert_celcius_to_fahrenheit(354));

#Task 5:
#Write a function called check_season, it takes a month parameter and returns the season:
Autumn, Winter, Spring, or Summer
def check_season(month):
    Summer = ['january', 'february', 'december']
    Autumn = ['september', 'october', 'november']
    Winter = ['march', 'april', 'may']
    Spring = ['june', 'july', 'august']
    month = str(month).lower();
    if(month in Summer):
        return "Summer";
    elif(month in Autumn):
        return "Autumn";
    elif(month in Winter):
        return "Winter";
```

```python
    elif(month in Spring):
        return "Spring";
    else:
        return "Invalid Month";

print("Task 5: January is in", check_season("January"), "season");

#Task 6:
#Write a function called calculate_slope which return the slope of a linear equation.
def calculate_slope(equation):
    # linear equation -> y = mx + b;
        #eq -> y = mx + b
    equation = equation.replace(" ", "");
    parts = equation.split("=");

    if(parts[0].strip() != 'y' or len(parts) != 2):
        return "Invalid Linear Equation : Expected format -- y = mx + b";

    parts[1] = parts[1].strip();
    index_of_x = parts[1].index('x');
    slope = parts[1][0:index_of_x];
    try:
        slope = float(slope);
        return slope;
    except ValueError as ve:
        return "Invalid Slope value is given! Try again!";

print("Task 6: The slope of linear equation (y = -532x + 67): ", calculate_slope("y = -532x +
67"));

#Task 7:
#Quadratic equation is calculated as follows: ax² + bx + c = 0. Write a function which
calculates solution set of a quadratic equation, solve_quadratic_eqn.
def solve_quadratic_eqn(equation):
    try:
        parts = equation.split("=")
        j = 0;
        for i in parts:
            parts[j] = i.strip()
            j += 1;

        if (int(parts[1]) != 0 and int(parts[1]) > 0):
            parts[0] += ' - ' + parts[1];
            parts[1] = '0';
        elif (int(parts[1]) != 0 and int(parts[1]) < 0):
            parts[0] += ' + ' + parts[1][1];
            parts[1] = '0';
        else:
            pass;

        parts[0] = parts[0].replace(" ", '');
        a = 0;
        if(parts[0][0:parts[0].index('x^2')] == '-'):
            a = -1;
        elif(parts[0][0:parts[0].index('x^2')] == ''):
            a = +1;
        elif (parts[0][0:parts[0].index('x^2')] != '-') and
(parts[0][0:parts[0].index('x^2')] != ''):
            a = int(parts[0][0:parts[0].index('x^2')])
        else:
            a = 0;

        b = 0;
        if(parts[0][(parts[0].index('^2')+1)+1 : parts[0].rindex('x')]) == '-':
            b = -1;
        elif(parts[0][(parts[0].index('^2')+1)+1 : parts[0].rindex('x')]) == '+':
            b = +1
        elif(parts[0][(parts[0].index('^2')+1)+1 : parts[0].rindex('x')]) != '-' or
(parts[0][(parts[0].index('^2')+1)+1 : parts[0].rindex('x')]) != '+' :
            b = int(parts[0][(parts[0].index('^2')+1)+1: parts[0].rindex('x')])
        else:
            b = 0;
        # c = int(parts[0][parts[0].rindex('x')+1: ]);
        c = 0;
        if(parts[0][parts[0].rindex('x')+1: ] == ''):
            c = 0;
        else:
            c = int(parts[0][parts[0].rindex('x')+1: ]);

        import math as m;
        discriminant = (b**2) - (4*a*c);
        if(discriminant > 0):
            print("Real Roots".upper());
            root_1 = (-(float(b)) + m.sqrt(discriminant)) / (2.0*a)
            root_2 = (-(float(b)) - m.sqrt(discriminant)) / (2.0*a)
            print("Equation : ", equation);
            print("|a =", a , "|b =", b, "|c =", c, "|");
            print("Solution Set : {", root_1, ",", root_2, "}");
        elif(discriminant == 0):
            print("Equal Roots".upper());
            roots = -float(b)/(2.0*a);
            print("Equation : ", equation);
            print("|a =", a , "|b =", b, "|c =", c, "|");
            print("Solution Set : {", roots, ",", roots, "}");
        else:
            print("Imaginary Roots".upper());
            print("Equation : ", equation);
            print("|a =", a , "|b =", b, "|c =", c, "|");
            print("Expensive Computation");
    except Exception as E:
        print("Error Occured : " , E, "\nOr else please enter the equation in format : ax^2 +
bx + c = 0/or/ax^2 + bx = c");
```

```python
print("Task 7 --> RESULT is as follows: ");
solve_quadratic_eqn("2x^2 - 4x = 9");

#Task 8:
#Declare a function named print_list. It takes a list as a parameter and it prints out each
element of the list.
def print_list(list_of_items):
    j = 1;
    for i in list_of_items:
        print("Item -", j , ": ", i);
        j += 1;

print("Task 8 --> RESULT is as follows");
print_list(['Hi', 'Hello', 'Vanakam', 'Adabhbarsey', 'Namasthe', 'Ciao']);

#Task 9:
#Declare a function named reverse_list. It takes an array as a parameter and it should return
the reverse of the array (use loop).
def reverse_list(list_of_items):
    reversed_list = [];
    for i in range(len(list_of_items)-1, -1, -1):
        reversed_list.append(list_of_items[i]);
    return reversed_list;

print("Task 9 : Actual list --> [1, 2, 3, 4, 5], Reversed List --> ", end = "");
print(reverse_list([1,2,3,4,5]));

#Task 10:
def capitalize_list_items(list_of_items):
    new_list = []
    for i in list_of_items:
        new_list.append(i.capitalize());
    return new_list;

print("Task 10 : Actual list --> ['ab cd', 'ef gh', 'ij kl'], Reversed List --> ", end = "");
print(capitalize_list_items(['ab cd', 'ef gh', 'ij kl']));

#Task 11:
#Declare a function named add_item. It takes a list and an item as parameter. It returns a
list with the item added at the end.
def add_item(list_of_items, item):
    list_of_items.append(item);
    return list_of_items;

print("Task 11: Actual List --> [1, 2, 3, 4, 5] Add:6, Added List --> ", end="");
print(add_item([1,2,3,4,5], 6));

#Task 12:
#Declare a function named remove_item. It takes a list and an item as paramter. It returns a
list with the item removed at the end.
```

```python
def removed_item(list_of_items, item):
    list_of_items.remove(item);
    return list_of_items;

print("Task 12: Actual List --> [1, 2, 3, 4, 5] Remove:4, Removed List --> ", end="");
print(removed_item([1,2,3,4,5], 4));

#Task 13:
def sum_of_numbers(upto_number):
    sum = 0;
    for i in range(upto_number+1):
        sum += i;
    return sum;

print("Task 13: Sum of numbers from 1 to 5:", sum_of_numbers(5));

#Task 14:
#Declare a function named sum_of_odds. It takes a number parameter and it adds all the odd numbers in that range.
def sum_of_odds(upto_number):
    sum = 0;
    for i in range(upto_number+1):
        if i%2 != 0:
            sum += i;
    return sum;

print("Task 14: Sum of odd numbers from 1 to 5:", sum_of_odds(5));

#Task 15:
#Declare a function named sum_of_evens. It takes a number parameter and it adds all the even numbers in that range.
def sum_of_evens(upto_number):
    sum = 0
    for i in range(upto_number+1):
        if(i%2 == 0):
            sum += i;
    return sum;

print("Task 15: Sum of even numbers from 1 to 5:", sum_of_evens(5));
```

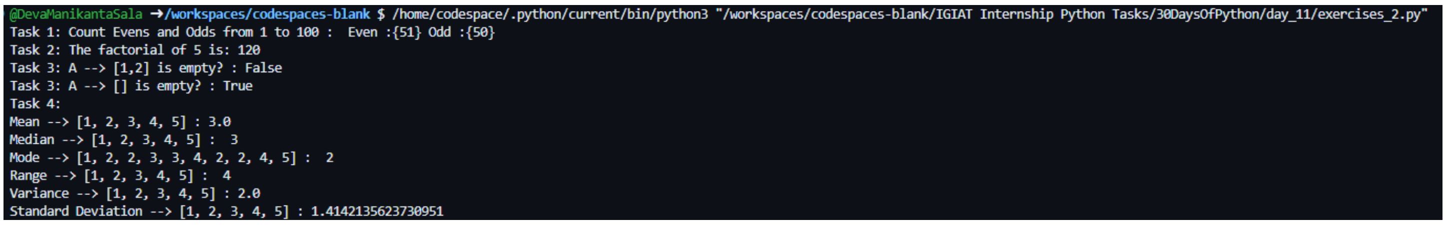
Outputs:


```
@DevaManikantaSala ➜/workspaces/codespaces-blank $ /home/codespace/.python/current/bin/python3 "/workspaces/codespaces-blank/IGIAT Internship Python Tasks/30DaysOfPython/day_11/exercises_1.py"
 Task 1: Sum of 1 and 2 :  3
 Task 2: Radius 5: Area of Circle is :  78.53981633974483
 Task 3: Sum of 1, 2, 3:  6
 Task 3: Sum of 1, 'Deva', 3:  All values are not same type
 Task 4: 354°C in °F :  669.2
 Task 5: January is in Summer season
 Task 6: The slope of linear equation (y = -532x + 67):  -532.0
 Task 7 --> RESULT is as follows:
 REAL ROOTS
 Equation :  2x^2 - 4x = 9
 |a = 2 |b = -4 |c = -9 |
 Solution Set : { 3.345207879911715 , -1.3452078799117149 }
 Task 8 --> RESULT is as follows
 Item - 1 :  Hi
 Item - 2 :  Hello
 Item - 3 :  Vanakam
 Item - 4 :  Adabhbarsey
 Item - 5 :  Namasthe
 Item - 6 :  Ciao
 Task 9 : Actual list --> [1, 2, 3, 4, 5], Reversed List --> [5, 4, 3, 2, 1]
 Task 10 : Actual list --> ['ab cd', 'ef gh', 'ij kl'], Reversed List --> ['Ab cd', 'Ef gh', 'Ij kl']
 Task 11: Actual List --> [1, 2, 3, 4, 5] Add:6, Added List --> [1, 2, 3, 4, 5, 6]
 Task 12: Actual List --> [1, 2, 3, 4, 5] Remove:4, Removed List --> [1, 2, 3, 5]
 Task 13: Sum of numbers from 1 to 5: 15
 Task 14: Sum of odd numbers from 1 to 5: 9
 Task 15: Sum of even numbers from 1 to 5: 6
@DevaManikantaSala ➜/workspaces/codespaces-blank $
```

Exercise Level 2

```python
#Task 1:
#Declare a function named evens_and_odds. It takes a positive integer as a parameter and it counts number of even and odds from 1 to number.
def evens_and_odds(number):
    count_even, count_odd = 0, 0
    for i in range(number+1):
        if(i % 2 == 0):
            count_even += 1;
        else:
            count_odd += 1;
    return str("Even :{" + str(count_even) + "}" + " Odd :{" + str(count_odd) + "}");
print("Task 1: Count Evens and Odds from 1 to 100 : ", evens_and_odds(100));

#Task 2:
#Call your function factorial, it takes a whole number as a parameter and it return factorial of the number
def factorial(number):
    if (number == 1):
        return 1;
    elif (number == 0):
        return 0;
    else:
        return number * (factorial(number-1));

print("Task 2: The factorial of 5 is:", factorial(5));

#Task 3:
#Call your function is_empty, it takes a parameter and it checks if it is empty or not
def is_empty(variable):
    if variable is None or len(variable) == 0:
        return True;
    else:
        return False;

print("Task 3: A --> [1,2] is empty? :", is_empty([1,2]));
print("Task 3: A --> [] is empty? :", is_empty([]));

#Task 4:
''' Write different functions which take lists.
    They should calculate_mean, calculate_median, calculate_mode, calculate_range,
calculate_variance, calculate_std (standard deviation).
'''
def calculate_mean(values_list):
    mean = sum(values_list)/len(values_list);
    return mean;

def calculate_median(values_list):
    values_list.sort();
    median = 0;
    if (len(values_list) % 2 == 1):
        median = values_list[len(values_list)//2];
    else:
        median = (values_list[len(values_list)//2 - 1] + values_list[len(values_list)//2])
/2;

    return median;

def calculate_mode(values_list):
    counts = [];
    for i in values_list:
        counts.append(values_list.count(i));
    counts.sort();
    counts.reverse();
    for i in values_list:
        if(counts[0] == values_list.count(i)):
            return i;

def calculate_range(values_list):
    range = max(values_list) - min(values_list);
    return range;

def calculate_variance(values_list):
    mean = calculate_mean(values_list);
    variance = sum((x - mean)**2 for x in values_list) / len(values_list);
    return variance;

import math as m
def calculate_std(values_list):
    standard_deviation = m.sqrt(calculate_variance(values_list));
    return standard_deviation;

print("Task 4:");
print("Mean --> [1, 2, 3, 4, 5] :", calculate_mean([1, 2, 3, 4, 5]));
print("Median --> [1, 2, 3, 4, 5] : ", calculate_median([1, 2, 3, 4, 5]));
print("Mode --> [1, 2, 2, 3, 3, 4, 2, 2, 4, 5] : ", calculate_mode([1, 2, 2, 3, 3, 4, 2, 2,
4, 5]));
print("Range --> [1, 2, 3, 4, 5] : ", calculate_range([1, 2, 3, 4, 5]));
print("Variance --> [1, 2, 3, 4, 5] :", calculate_variance([1, 2, 3, 4, 5]));
print("Standard Deviation --> [1, 2, 3, 4, 5] :", calculate_std([1, 2, 3, 4, 5]));
```

Output:



```
@DevaManikantaSala ➜/workspaces/codespaces-blank $ /home/codespace/.python/current/bin/python3 "/workspaces/codespaces-blank/IGIAT Internship Python Tasks/30DaysOfPython/day_11/exercises_2.py"
Task 1: Count Evens and Odds from 1 to 100 :  Even :{51} Odd :{50}
Task 2: The factorial of 5 is: 120
Task 3: A --> [1,2] is empty? : False
Task 3: A --> [] is empty? : True
Task 4:
Mean --> [1, 2, 3, 4, 5] : 3.0
Median --> [1, 2, 3, 4, 5] :  3
Mode --> [1, 2, 2, 3, 3, 4, 2, 2, 4, 5] :  2
Range --> [1, 2, 3, 4, 5] :  4
Variance --> [1, 2, 3, 4, 5] : 2.0
Standard Deviation --> [1, 2, 3, 4, 5] : 1.4142135623730951
```

Exercise Level 3

Step 1: Create a folder 'data' and download 'countries.py' click on this link to download the file and paste it in 'data' folder.
Try this link if it doesn't download:
https://drive.google.com/uc?export=download&id=19FKmSln0zclQ97yUkFponmuWvyO4OGe9

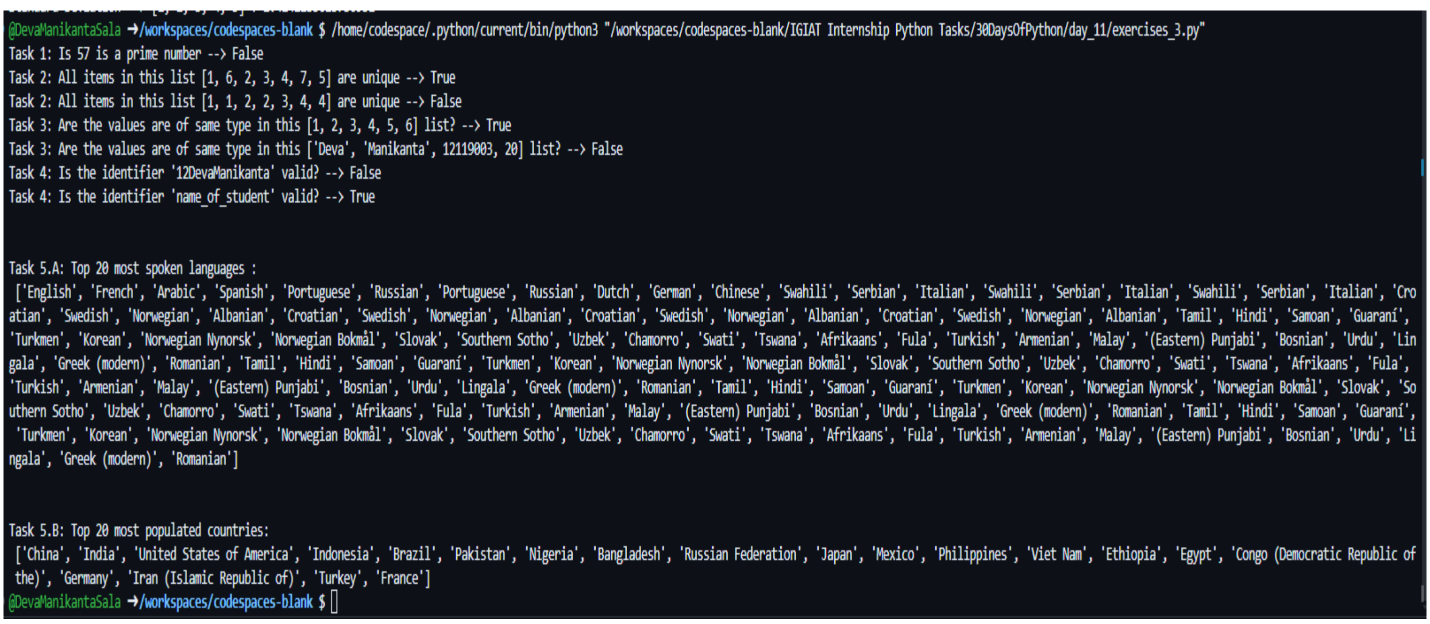
Step 2: Now, download this file and paste it in 'data' folder 'countries_data.py' click on this link to download the file.

Try this link if it doesn't download:
https://drive.google.com/uc?export=download&id=1aumrIQiumLBau9hTUPNEQBYgxZp-MwAo

```python
#Task 1:
#Write a function called is_prime, which checks if a number is prime
def is_prime(number):
    if(number < 2):
        return False;
    i = 2;
    import math as m;
    while(i <= int(m.sqrt(number))):
        if (number % i == 0):
            return False;
        i += 1;
    return True;

print("Task 1: Is 57 is a prime number -->", is_prime(57));

#Task 2:
#Write a function which checks if all items are unique in the list.
def is_all_items_unique(values_list):
    for i in values_list:
        if (i in values_list) and (values_list.count(i) > 1):
            return False;
    return True;

print("Task 2: All items in this list [1, 6, 2, 3, 4, 7, 5] are unique -->",
is_all_items_unique([1, 6, 2, 3, 4, 7, 5]));
```

```python
print("Task 2: All items in this list [1, 1, 2, 2, 3, 4, 4] are unique -->",
is_all_items_unique([1, 1, 2, 2, 3, 4, 4]));

#Task 3:
#Write a function which checks if all the items of the list are of the same data type.
def is_all_items_same_type(values_list):
    for i in values_list:
        if not isinstance(i, type(values_list[0])):
            return False
            break;
    return True;

print("Task 3: Are the values are of same type in this [1, 2, 3, 4, 5, 6] list? -->",
is_all_items_same_type([1, 2, 3, 4, 5, 6]));
print("Task 3: Are the values are of same type in this [\'Deva\', \'Manikanta\', 12119003,
20] list? -->", is_all_items_same_type(['Deva', 'Manikanta', 12119003, 20]));

#Task 4:
#Write a function which check if provided variable is a valid python variable.
def is_identifier(string):
    if(string.isidentifier()):
        return True;
    else:
        return False;
print("Task 4: Is the identifier \'12DevaManikanta\' valid? -->",
is_identifier("12DevaManikanta"));
print("Task 4: Is the identifier \'name_of_student\' valid? -->",
is_identifier("name_of_student"));

#Task 5:
#A) Create a function called the most_spoken_languages in the world. It should return 10 or
20 most spoken languages in the world in descending order
from data import countries_data as cd;
def most_spoken_languages():
    languages_spoken = []
    for item in cd.countries_data:
        for language in item['languages']:
            languages_spoken.append(language);

    unique_languages = set(languages_spoken);
    counts_of_unique_languages = {}
    for language in unique_languages:
        counts_of_unique_languages[language] = languages_spoken.count(language);
    counts = list(counts_of_unique_languages.values());
    counts.sort();
    counts.reverse();
    counts = counts[0:20];
    i = 0;
    twenty_most_spoken_languages = []
    while(i < len(counts)):
        for language, count in counts_of_unique_languages.items():
            if(count == counts[i]):
                twenty_most_spoken_languages.append(language);
        i += 1;
    return twenty_most_spoken_languages;
print("\n\nTask 5.A: Top 20 most spoken languages : \n", most_spoken_languages());

#B) Create a function called the most_populated_countries. It should return 10 or 20 most
populated countries in descending order.
from data import countries_data as cd;
def most_populated_countries():
    population_of_countries = []
    for item in cd.countries_data:
        population_of_countries.append(item['population'])

    population_of_countries.sort();
    population_of_countries.reverse();
    population_of_countries = population_of_countries[0:20];

    i = 0;
    twenty_most_populated_countries = [];
    while(i < len(population_of_countries)):
        for item in cd.countries_data:
            if(item['population'] == population_of_countries[i]):
                twenty_most_populated_countries.append(item['name']);
        i += 1;
    return twenty_most_populated_countries;
print("\n\nTask 5.B: Top 20 most populated countries:\n", most_populated_countries());
```

Output:



```
@DevaManikantaSala →/workspaces/codespaces-blank $ /home/codespace/.python/current/bin/python3 "/workspaces/codespaces-blank/IGIAT Internship Python Tasks/30DaysOfPython/day_11/exercises_3.py"
Task 1: Is 57 is a prime number --> False
Task 2: All items in this list [1, 6, 2, 3, 4, 7, 5] are unique --> True
Task 2: All items in this list [1, 1, 2, 2, 3, 4, 4] are unique --> False
Task 3: Are the values are of same type in this [1, 2, 3, 4, 5, 6] list? --> True
Task 3: Are the values are of same type in this ['Deva', 'Manikanta', 12119003, 20] list? --> False
Task 4: Is the identifier '12DevaManikanta' valid? --> False
Task 4: Is the identifier 'name_of_student' valid? --> True


Task 5.A: Top 20 most spoken languages :
 ['English', 'French', 'Arabic', 'Spanish', 'Portuguese', 'Russian', 'Portuguese', 'Russian', 'Dutch', 'German', 'Chinese', 'Swahili', 'Serbian', 'Italian', 'Swahili', 'Serbian', 'Italian', 'Swahili', 'Serbian', 'Italian', 'Cro
atian', 'Swedish', 'Norwegian', 'Albanian', 'Croatian', 'Swedish', 'Norwegian', 'Albanian', 'Croatian', 'Swedish', 'Norwegian', 'Albanian', 'Croatian', 'Swedish', 'Norwegian', 'Albanian', 'Tamil', 'Hindi', 'Samoan', 'Guaraní',
'Turkmen', 'Korean', 'Norwegian Nynorsk', 'Norwegian Bokmål', 'Slovak', 'Southern Sotho', 'Uzbek', 'Chamorro', 'Swati', 'Tswana', 'Afrikaans', 'Fula', 'Turkish', 'Armenian', 'Malay', '(Eastern) Punjabi', 'Bosnian', 'Urdu', 'Lin
gala', 'Greek (modern)', 'Romanian', 'Tamil', 'Hindi', 'Samoan', 'Guaraní', 'Turkmen', 'Korean', 'Norwegian Nynorsk', 'Norwegian Bokmål', 'Slovak', 'Southern Sotho', 'Uzbek', 'Chamorro', 'Swati', 'Tswana', 'Afrikaans', 'Fula',
'Turkish', 'Armenian', 'Malay', '(Eastern) Punjabi', 'Bosnian', 'Urdu', 'Lingala', 'Greek (modern)', 'Romanian', 'Tamil', 'Hindi', 'Samoan', 'Guaraní', 'Turkmen', 'Korean', 'Norwegian Nynorsk', 'Norwegian Bokmål', 'Slovak', 'So
uthern Sotho', 'Uzbek', 'Chamorro', 'Swati', 'Tswana', 'Afrikaans', 'Fula', 'Turkish', 'Armenian', 'Malay', '(Eastern) Punjabi', 'Bosnian', 'Urdu', 'Lingala', 'Greek (modern)', 'Romanian', 'Tamil', 'Hindi', 'Samoan', 'Guaraní',
 'Turkmen', 'Korean', 'Norwegian Nynorsk', 'Norwegian Bokmål', 'Slovak', 'Southern Sotho', 'Uzbek', 'Chamorro', 'Swati', 'Tswana', 'Afrikaans', 'Fula', 'Turkish', 'Armenian', 'Malay', '(Eastern) Punjabi', 'Bosnian', 'Urdu', 'Li
ngala', 'Greek (modern)', 'Romanian']


Task 5.B: Top 20 most populated countries:
 ['China', 'India', 'United States of America', 'Indonesia', 'Brazil', 'Pakistan', 'Nigeria', 'Bangladesh', 'Russian Federation', 'Japan', 'Mexico', 'Philippines', 'Viet Nam', 'Ethiopia', 'Egypt', 'Congo (Democratic Republic of
 the)', 'Germany', 'Iran (Islamic Republic of)', 'Turkey', 'France']
@DevaManikantaSala →/workspaces/codespaces-blank $ 
```