**Fundamentals of Python**

# Day -1

1. Write a code to find the minimum among three given numbers.
2. Write a code to check whether a given number is a palindrome.
3. Write a code to find the sum of numbers divisible by 4.The code must allow the user to accept a number and add it to the sum if it is divisible by 4. It should continue accepting numbers as long as the user wants to provide an input and should display the final sum.
4. A three digit number is said to be an "Armstrong number" if the sum of the third power of its individual digits is equal to the number itself.Write a program to check whether a number is armstrong or not.

5. JIT University offering degree courses to students has decided to provide scholarship based on the following details:

| Branch of study | Score (%) | Scholarship % | Remarks |
|---|---|---|---|
| Arts | Score is at least 90 | 50 | The student is eligible only for one scholarship% even if both the score conditions are valid for the given branch of study. In such cases, students are eligible for the highest scholarship% applicable among the two. |
| Arts | Score is an odd number | 5 | |
| Engineering | Score is more than 85 | 50 | |
| Engineering | Score is divisible by 7 | 5 | |

If there are 500 students who have joined the university, write a code to calculate and display the final fees to be paid by each student.
You may accept the branch of study, score and course fee as inputs for each student and calculate the final fees to be paid by each student based on formulae given below:
Scholarship amount=course fee * (scholarship%)
Final fee= course fee - scholarship amount

6. The flight ticket rates for a round-trip (Mumbai->Dubai) were as follows:
    - Rate per Adult: Rs. 37550.0
    - Rate per Child: 1/3rd of the rate per adult
    - Service Tax: 7% of the ticket amount (including all passengers)
    - As it was a holiday season, the airline also offered a 10% discount on the final ticket cost (after inclusion of the service tax).
    - Find and display the total ticket cost for a group which had adults and children.

**Day-2**

7. Write a python program that displays a message as follows for a given number:
    - If it is a multiple of three, display "Zip"
    - If it is a multiple of five, display "Zap".
    - If it is a multiple of both three and five, display "Zoom".
    - If it does not satisfy any of the above given conditions, display "Invalid".

8. A teacher in a school wants to find and display the grade of a student based on his/her percentage score. The criterion for grades is as given below:

| Score (both inclusive) | Grade |
|---|---|
| Between 80 and 100 | A |
| Between 73 and 79 | B |
| Between 65 and 72 | C |

| | |
|---|---|
| Between 0 and 64 | D |
| Any other value | Z |

Assume that the percentage score is a whole number. Write a python program for the above requirement.

9. Write a python program to find and display the product of three positive integer values based on the rule mentioned below:

It should display the product of the three values except when one of the integer value is 7. In that case, 7 should not be included in the product and the values to its left also should not be included.

If there is only one value to be considered, display that value itself. If no values can be included in the product, display -1.

Note: Assume that if 7 is one of the positive integer values, then it will occur only once. Refer the sample I/O given below.

| Sample Input | Expected Output |
|---|---|
| 1, 5, 3 | 15 |
| 3, 7, 8 | 8 |
| 7, 4, 3 | 12 |
| 1, 5, 7 | -1 |

10. You have x no. of 5 rupee coins and y no. of 1 rupee coins. You want to purchase an item for amount z. The shopkeeper wants you to provide exact change. You want to pay using minimum number of coins. How many 5 rupee coins and 1 rupee coins will you use? If exact change is not possible then display -1.

| Sample Input | | | Expected Output | |
|---|---|---|---|---|
| Available Rs. 1 coins | Available Rs. 5 notes | Amount to be made | Rs. 1 coins needed | Rs. 5 notes needed |
| 2 | 4 | 21 | 1 | 4 |
| 11 | 2 | 11 | 1 | 2 |
| 3 | 3 | 19 | -1 | |

**Day -3**

11. Write a Python program to generate the next 15 leap years starting from a given year. Populate the leap years into a list and display the list.

12. ARS Gems Store sells different varieties of gems to its customers.

Write a Python program to calculate the bill amount to be paid by a customer based on the list of gems and quantity purchased. Any purchase with a total bill amount above Rs.30000 is entitled to 5% discount. If any gem required by the customer is not available in the store, then consider total bill amount to be -1.

Assume that quantity required by the customer for any gem will always be greater than 0.

Perform case-sensitive comparison wherever applicable.

gems_list=["Emerald","Ivory","Jasper","Ruby","Garnet"]

#Price of gems available in the store. gems_list and price_list have one-to-one correspondence
price_list=[1760,2119,1599,3920,3999]

#List of gems required by the customer
reqd_gems=["Ivory","Emerald","Garnet"]

#Quantity of gems required by the customer. reqd_gems and reqd_quantity have one-to-one correspondence
reqd_quantity=[3,10,12]

13. Write a python function to check whether three given numbers can form the sides of a triangle.
Hint
: Three numbers can be the sides of a triangle if none of the numbers are greater than or equal to the sum of the other two numbers.

14. Write a python program to solve a classic ancient Chinese puzzle.
We count 35 heads and 94 legs among the chickens and rabbits in a farm. How many rabbits and how many chickens do we have?

| Sample Input | Expected Output |
|---|---|
| heads-150 legs-400 | 100 50 |
| heads-3 legs-11 | No solution |
| heads-3 legs-12 | 0 3 |
| heads-5 legs-10 | 5 0 |

15. Write a python program which finds the maximum number from num1 to num2 (num2 inclusive) based on the following rules.

1. Always num1 should be less than num2

2. Consider each number from num1 to num2 (num2 inclusive). Populate the number into a list, if the below conditions are satisfied
    1. Sum of the digits of the number is a multiple of 3
    2. Number has only two digits
    3. Number is a multiple of 5
3. Display the maximum element from the list

In case of any invalid data or if the list is empty, display -1.

16. Write a python program to generate the ticket numbers for specified number of passengers traveling in a flight as per the details mentioned below:
The ticket number should be generated as airline:src:dest:number

where

1. Consider AI as the value for airline
2. src and dest should be the first three characters of the source and destination cities.
3. number should be auto-generated starting from 101

The program should return the list of ticket numbers of last five passengers.
Note: If passenger count is less than 5, return the list of all generated ticket numbers.

| Sample Input | Expected Output |
|---|---|
| airline = AI<br>source = Bangalore<br>destination = London<br>no_of_passengers = 10 | ['AI:Ban:Lon:106', 'AI:Ban:Lon:107', 'AI:Ban:Lon:108', 'AI:Ban:Lon:109', 'AI:Ban:Lon:110'] |

| | |
|---|---|
| airline = BA<br>source = Australia<br>destination = France<br>no_of_passengers = 2 | ['BA:Aus:Fra:101', 'BA:Aus:Fra:102'] |

17. Represent a small bilingual (English-Swedish) glossary given below as a Python dictionary

{"merry":"god", "christmas":"jul", "and":"och", "happy":"gott", "new":"nytt", "year":"ar"}

and use it to translate your Christmas wishes from English into Swedish.

That is, write a python function translate() that accepts the bilingual dictionary and a list of English words (your Christmas wish) and returns a list of equivalent Swedish words.

**Day-4**

18. The road transport corporation (RTC) of a city wants to know whether a particular bus-route is running on profit or loss.

Assume that the following information are given:
Price per litre of fuel = 70
Mileage of the bus in km/litre of fuel = 10
Price(Rs) per ticket = 80

The bus runs on multiple routes having different distance in kms and number of passengers. Write a function to calculate and return the profit earned (Rs) in each route. Return -1 in case of loss.

19. Given a string containing uppercase characters (A-Z), compress the string using Run Length encoding. Repetition of character has to be replaced by storing the length of that run.

Write a python function which performs the run length encoding for a given String and returns the run length encoded String.

Provide different String values and test your program.

| Sample Input | Expected Output |
|---|---|
| AAAABBBBCCCCCCCC | 4A4B8C |
| AABCCA | 2A1B2C1A |

20. Write a function, check_palindrome() to check whether the given string is a palindrome or not. The function should return true if it is a palindrome else it should return false.

Note: Initialize the string with various values and test your program. Assume that all the letters in the given string are all of the same case. Example: MAN, civic, WOW etc.

21. Care hospital wants to know the medical speciality visited by the maximum number of patients. Assume that the patient id of the patient along with the medical speciality visited by the patient is stored in a list. The details of the medical specialities are stored in a dictionary as follows:

```
{
"P":"Pediatrics",
"O":"Orthopedics",
"E":"ENT
}
```

Write a function to find the medical speciality visited by the maximum number of patients and return the name of the speciality.

Note:

1. Assume that there is always only one medical speciality which is visited by maximum number of patients.
2. Perform case sensitive string comparison wherever necessary.

| Sample Input | Expected Output |
|---|---|
| [101,P,102,O,302,P,305,P] | Pediatrics |
| [101,O,102,O,302,P,305,E,401,O,656,O] | Orthopedics |
| [101,O,102,E,302,P,305,P,401,E,656,O,987,E] | ENT |

**Day - 5**

22. Write a python function, find_pairs_of_numbers() which accepts a list of positive integers with no repetitions and returns count of pairs of numbers in the list that adds up to n. The function should return 0, if no such pairs are found in the list.

| Sample Input | Expected Output |
|---|---|
| [1, 2, 7, 4, 5, 6, 0, 3], 6 | 3 |
| [3, 4, 1, 8, 5, 9, 0, 6], 9 | 4 |

23. A teacher is in the process of generating a few reports based on the marks scored by the students of her class in a project based assessment.
Assume that the marks of her 10 students are available in a tuple. The marks are out of 25.

Write a python program to implement the following functions:

1. find_more_than_average(): Find and return the percentage of students who have scored more than the average mark of the class.
2. generate_frequency(): Find how many students have scored the same marks. For example, how many have scored 0, how many have scored 1, how many have scored 3….how many have scored 25. The result should be populated in a list and returned.
3. sort_marks(): Sort the marks in the increasing order from 0 to 25. The sorted values should be populated in a list and returned.

| Sample Input | Expected Output |
|---|---|
| list_of_marks = (12,18,25,24,2,5,18,20,20,21) | 70.0<br>[0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2, 0, 2, 1, 0, 0, 1, 1]<br>[2, 5, 12, 18, 18, 20, 20, 21, 24, 25] |

24. Write a python function, create_largest_number(), which accepts a list of numbers and returns the largest number possible by concatenating the list of numbers.

Note: Assume that all the numbers are two digit numbers.

| Sample Input | Expected Output |
|---|---|
| 23,34,55 | 553423 |

25. Write a python function, check_double(number) which accepts a whole number and returns True if it satisfies the given conditions.

1. The number and its double should have exactly the same number of digits.
2. Both the numbers should have the same digits ,but in different order.

Otherwise it should return False.

Example: If the number is 125874 and its double, 251748, contain exactly the same digits, but in a different order.

26. A vendor at a food court is in the process of automating his order management system. The vendor serves the following menu – Veg Roll, Noodles, Fried Rice and Soup and also maintains the quantity available for each item. The customer can order any combination of items. The customer is provided the item if the requested quantity of item is available with the vendor.

Write a python program which implements the following functions.

place_order(*item_tuple): This function accepts the order placed by the customer. Consider it to be a variable length argument as each customer may have a different order.

The function should check whether the items requested are present in the vendor's menu and if so, it should check whether the requested quantity is available for each by invoking the check_quantity_available() method.

The function should display appropriate messages for each item in the order for the below scenarios:

1. When the requested item is not available in vendor's menu, display <Item Name> is not available
2. When the quantity requested by the customer is not available, display <Item Name> stock is over
3. When the requested quantity of the item is available with the vendor, display <Item Name> is available

check_quantity_available(index,quantity_requested): This function should check whether the requested quantity of the specified item is available. If so, it should reduce the quantity requested from the quantity available for that item and return True. Otherwise, it should return False.

Test your code by using the given sample inputs.
Verify your code by using the 2nd sample input(highlighted) given below:

| Sample Input | | Expected Output |
|---|---|---|
| Menu and quantity available | Items Ordered | |
| (Veg Roll, Noodles, Fried Rice , Soup)<br>[2,200,250,3] | Veg Roll,2<br>Noodles,2 | Veg Roll is available<br>Noodles is available |
| (Veg Roll, Noodles, Fried Rice , Soup)<br>[2,200,3,0] | Fried Rice,2<br>Soup,1 | |

**Day-6**

27. Write a recursive function, is_palindrome() to find out whether a string is a palindrome or not. The function should return true, if it is a palindrome. Else it should return false.

Note- Perform case insensitive operations wherever necessary.

28. A 10-substring of a number is a substring of its digits that sum up to 10.

For example, the 10-substrings of the number 3523014 are:
352<u>3014</u>, 3<u>523</u>014, 3<u>5230</u>14, 35<u>23014</u>

Write a python function, find_ten_substring(num_str) which accepts a string and returns the list of 10-substrings of that string.

Handle the possible errors in the code written inside the function.

| Sample Input | Expected Output |
|---|---|
| '3523014' | ['5230', '23014', '523', '352'] |

29. Given a number n, write a program to find the sum of the largest prime factors of each of nine consecutive numbers starting from n.
g(n) = f(n) + f(n+1) + f(n+2) + f(n+3) + f(n+4) + f(n+5) + f(n+6) + f(n+7) + f(n+8)
where, g(n) is the sum and f(n) is the largest prime factor of n

For example,
g(10)=f(10)+f(11)+f(12)+f(13)+f(14)+f(15)+f(16)+f(17)+f(18)
=5 + 11 + 3 + 13 + 7 + 5 + 2 + 17 + 3
=66

30. Write a python function find_smallest_number() which accepts a number n and returns the smallest number having n divisors.

| Sample Input | Expected Output |
|---|---|
| 16 | 120 |

31. Write a python function find_duplicates(), which accepts a list of numbers and returns another list containing all the duplicate values in the input list. If there are no duplicate values, it should return an empty list.

| Sample Input | Expected Output |
|---|---|
| [12,54,68,759,24,15,12,68,987,758,25,69] | [12, 68] |

32. The below function is written to check whether a given three digit number is an Armstrong number.

Hint: An “Armstrong number” is an n-digit number that is equal to the sum of the nth powers of its individual digits.
Example: 371 is an Armstrong number as 371 = 3^3 +7^3+ 1^3

**Day - 7**

33. Write a python function, nearest_palindrome() which accepts a number and returns the nearest palindrome greater than the given number.

| Sample Input | Expected Output |
|---|---|
| 12300 | 12321 |
| 12331 | 12421 |

34. Write a python function, encrypt_sentence() which accepts a message and encrypts it based on rules given below and returns the encrypted message.

Words at odd position -> Reverse It

Words at even position -> Rearrange the characters so that all consonants appear before the vowels and their order should not change

Note:

1. Assume that the sentence would begin with a word and there will be only a single space between the words.
2. Perform case sensitive string operations wherever necessary.

Also write the pytest test cases to test the program.

| Sample Input | Expected Output |
|---|---|

| | |
|---|---|
| the sun rises in the east | eht snu sesir ni eht stea |

Estimated Time: 40 minutes

35. Write a python function, find_correct() which accepts a dictionary and returns a list as per the rules mentioned below.
The input dictionary will contain correct spelling of a word as key and the spelling provided by a contestant as the value.

The function should identify the degree of correctness as mentioned below:
CORRECT, if it is an exact match

ALMOST CORRECT, if no more than 2 letters are wrong
WRONG, if more than 2 letters are wrong or if length (correct spelling versus spelling given by contestant) mismatches.

and return a list containing the number of CORRECT answers, number of ALMOST CORRECT answers and number of WRONG answers.
Assume that the words contain only uppercase letters and the maximum word length is 10.

Also write the pytest test cases to test the program.

| Sample Input | Expected Output |
|---|---|
| {"THEIR": "THEIR", "BUSINESS": "BISINESS","WINDOWS":"WINDMILL","WERE":"WEAR","SAMPLE":"SAMPLE"} | [2, 2, 1] |

Estimated Time: 40 minutes

36. In a fair coin we have an equal chance (50%) of either getting a ‘head’ or ‘tail’.  That is if we toss the coin a large number of times we would observe head approximately 50% of the time. Write a program to implement a biased coin toss where the chance of getting a head is 70% (and tail 30%). That is if we invoke the program 1000 times we should see the head randomly approximately 700 times.

37. Write python function, sms_encoding() which accepts a sentence and converts it into an abbreviated sentence to be sent as SMS and returns the abbreviated sentence.

Rules are as follows:

a. Spaces are to be retained as is

b. Each word should be encoded separately

- If a word has only vowels then retain the word as is
- If a word has a consonant (at least 1) then retain only those consonants

Note: Assume that the sentence would begin with a word and there will be only a single space between the words.

| Sample Input | Expected Output |
|---|---|
| I love Python | I lv Pythn |
| MSD says I love cricket and tennis too | MSD sys I lv crckt nd tnns t |
| I will not repeat mistakes | I wll nt rpt mstks |

Estimated time: 45 minutes

38. Consider sample data as follows: sample_data=range(1,11)

Create two functions: odd() and even()

The function even() returns a list of all the even numbers from sample_data

The function odd() returns a list of all the odd numbers from sample_data

Create a function sum_of_numbers() which will accept the sample data and/or a function.
If a function is not passed, the sum_of_numbers() should return the sum of all the numbers from sample_data
If a function is passed, the sum_of_numbers() should return the sum of numbers returned from the function passed.

**Day - 8**

39. Write a python function, check_anagram() which accepts two strings and returns True, if one string is an anagram of another string. Otherwise returns False.

The two strings are considered to be an anagram if they contain repeating characters but none of the characters repeat at the same position. The length of the strings should be the same.

Note: Perform case insensitive comparison wherever applicable.

| Sample Input | Expected Output |
|---|---|
| eat, tea | True |
| backward,drawback | False<br>(Reason: character 'a' repeats at position 6, not an anagram) |
| Reductions,discounter | True |
| About, table | False |

40. Write a python program that accepts a text and displays a string which contains the word with the largest frequency in the text and the frequency itself separated by a space.

Rules:

1. The word should have the largest frequency.
2. In case multiple words have the same frequency, then choose the word that has the maximum length.

Assumptions:

1. The text has no special characters other than space.
2. The text would begin with a word and there will be only a single space between the words.

Perform case insensitive string comparisons wherever necessary.

| Sample Input | Expected Output |
|---|---|
| "Work like you do not need money love like you have never been hurt and dance like no one is watching" | like 3 |
| "Courage is not the absence of fear but rather the judgement that something else is more important than fear" | fear 2 |

**Day - 9**

41. The number, 197, is called a circular prime because all rotations of the digits: 197, 971, and 719, are themselves prime.

There are thirteen such primes below 100: 2, 3, 5, 7, 11, 13, 17, 31, 37, 71, 73, 79, and 97.

Write a python program to find and display the number of circular primes less than the given limit.

42. Use Luhn algorithm to validate a credit card number.

A credit card number has 16 digits, the last digit being the check digit. A credit card number can be validated using Luhn algorithm as follows:

Step 1a: From the second last digit (inclusive), double the value of every second digit.
Suppose the credit card number is 1456734512345698.
Take the double of 9,5,3,1,4,7,5 and 1
i.e., 18, 10, 6, 2, 8, 14, 10 and 2

Note: If any number is greater than 9, then sum the digits of that number.
i.e., 9, 1, 6, 2, 8, 5 ,1 and 2

Step 1b: Sum the numbers obtained in Step 1a.
i.e., 34

Step 2: Sum the remaining digits in the credit card and add it with the sum from Step 1b.
i.e., 34 + (8+6+4+2+5+3+6+4) = 72

Step 3: If the total is divisible by 10 then the number is valid else it is not valid.

Write a function, validate_credit_card_number(), which accepts a 16 digit credit card number and returns true if it is valid as per Luhn's algorithm or false, if it is invalid.

43. Write a python function to check whether the given number is a perfect number or not. The function should returns true if the number is a perfect number, else it should returns false.

Hint: Perfect number is a positive whole number that is equal to the sum of its proper divisors. The first perfect number is 6 as the sum of its proper positive divisors, 1,2 and 3 is 6. Other perfect numbers are 28, 496, 8128 ...

Extend the program written for the above problem to write another function to find all perfect numbers in a given list of numbers. Populate the perfect numbers in a list and return the list. If no perfect numbers are found, return an empty list.

Note- Reuse the code wherever possible.

44. Write a python function, remove_duplicates() which accepts a string and removes all duplicate characters from the given string and return it.

| Sample Input | Expected Output |
| --- | --- |
| 1122334455ababzzz@@@123#*#* | 12345abz@#* |

**Object Oriented Programming**

# Day 1

49. TechWorld, a technology training center, wants to allocate courses for instructors.
An instructor is identified by name, technology skills, experience and average feedback.

An instructor is allocated a course, if he/she satisfies the below two conditions:

- eligibility criteria:
  - if experience is more than 3 years, average feedback should be 4.5 or more
  - if experience is 3 years or less, average feedback should be 4 or more
- he/she should posses the technology skill for the course

Identify the class name and attributes to represent instructors.

Write a Python program to implement the class chosen with its attributes and methods.

**Note**:

1. Consider all instance variables to be private and methods to be public
2. An instructor may have multiple technology skills, so consider instance variable, technology_skill to be a list
3. **check_eligibility():** Return true if eligibility criteria is satisfied by the instructor. Else, return false
4. **allocate_course(technology):** Return true if the course which requires the given technology can be allocated to the instructor. Else, return false
5. Perform case sensitive string comparison

Represent a few objects of the class, initialize instance variables using setter methods, invoke appropriate methods and test your program.

50. A university wants to automate their admission process. Students are admitted based on the marks scored in the qualifying exam.

A student is identified by student id, age and marks in qualifying exam. Data are valid, if:

- Age is greater than 20
- Marks is between 0 and 100 (both inclusive)

A student qualifies for admission, if

- Age and marks are valid and
- Marks is 65 or more

Write a python program to represent the students seeking admission in the university.

The details of student class are given below.

**Class name:** Student

| | | |
|---|---|---|
| **Attributes (private)** | student_id<br>marks<br>age | |
| **Methods (public)** | __init__() | Create and initialize all instance variables to None |
| | validate_marks() | If data is valid, return true. Else, return false |
| | validate_age() | |
| | check_qualification() | Validate marks and age.<br>• If valid, check if marks is 65 or more.<br>  ○ If so return true<br>  ○ Else return false<br>• Else return false |
| | setter methods | Include setter methods for all instance variables to set its values |

| | | |
|---|---|---|
| | getter methods | Include getter methods for all instance variables to get its values |

# Day 2

51. Royal Orchid is a florist. They want to be alerted when stock of a flower goes below a particular level.
The flowers are identified using name, price per kg and stock available (in kgs).
Write a Python program to implement the above requirement.
Details of Flower class are given below:

**Class name:** Flower

| | | |
|---|---|---|
| **Attributes (private)** | flower_name<br>price_per_kg<br>stock_available | |
| **Methods (public)** | __init__() | Create and initialize all instance variables to None |
| | validate_flower() | Return true, if flower name is valid. Else, return false<br>(Refer table for valid flower names) |
| | validate_stock(required_quantity) | Accept the quantity required. Return true, if stock is available.<br>Else return false. |

| | | |
|---|---|---|
| | sell_flower(required_quantity)) | Accept the quantity required.<br>Validate flower name and stock.<br>If both are valid, update stock available based on the quantity required |
| | check_level() | Check if available stock is below the order level<br>If so, return true. Else, return false<br>(Refer table for order level of each flower) |
| | setter methods | Include setter methods for all instance variables to set its values |
| | getter methods | Include getter methods for all instance variables to get its values |

| **Flower Name** | **Level(in Kgs)** |
|---|---|
| Orchid | 15 |
| Rose | 25 |
| Jasmine | 40 |

**Note:** Perform case insensitive string comparison

Represent few flowers, initialize instance variables using setter methods, invoke appropriate methods and test your program.

52. Write a python program to find out if a given classroom is present in the left wing of a university building. Implement the class, Classroom given below.

| Classroom |
| --- |
| + classroom_list -> static |
| + search_classroom(class_room) -> static |

Method/Attribute description

1. classroom_list: Static list which store the name of the class rooms in the left wing
2. search_classroom(class_room): Static method which search for the given class room in the classroom_list. If found, return "Found". Else, return -1

Note: Perform case insensitive string comparison

For testing:

- Invoke search_classroom(class_room) static method on class, Classroom by passing the name of the class room to be searched
- Display appropriate message based on the return value of search_classroom(class_room)

# Day 3

53. Animal Welfare Trust is on a visit to the circus camp to have a look at the four talking parrots added to the camp.

A parrot is identified by its name and color. Apart from this, the trust has asked to assign a unique number for each parrot. The unique number should begin with 7001 and should be auto-incremented by 1 for every new parrot added to the camp.

Identify the class name and attributes so as to represent parrots from the list given.

Write a Python program to implement the class chosen with its attributes and methods. Represent few parrots, display their names, color and unique number.

**Note:** Consider all the attributes to be private and methods to be public. Include getter methods for all the instance variables.

54. Write a Python program to generate tickets for online bus booking, based on the class diagram given below.


**Method description:**

1. Initialize static variable counter to 0
2. **validate_source_destination():** Validate source and destination. source must always be Delhi and destination can be either Mumbai, Chennai, Pune or Kolkata. If both are valid, return true. Else return false
3. generate_ticket():
    1. Validate source and destination
    2. If valid, generate ticket id and assign it to attribute, ticket_id
       Ticket id should be generated with the first letter of source followed by first letter of destination and an auto-generated value starting from 01 (Ex: DM01, DP02,.. ,DK10,DC11)
    3. Else, set ticket_id as None

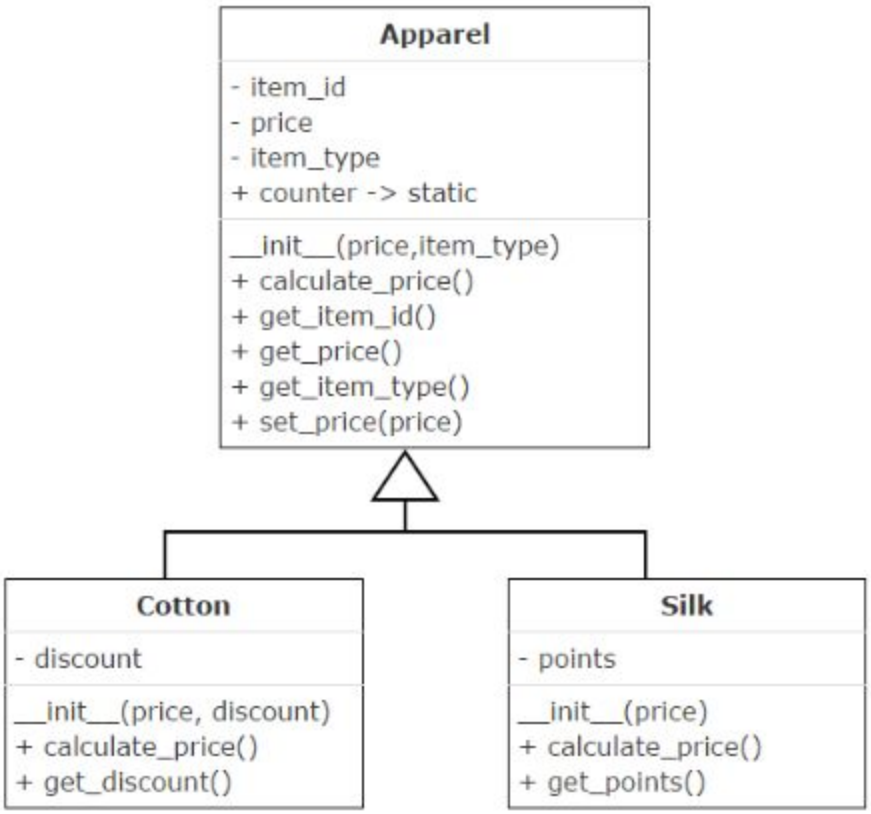
**Note:** Perform case insensitive string comparison

For testing:

- Create objects of Ticket class
- Invoke generate_ticket() method on Ticket object
- Display ticket id, passenger name, source, destination
- In case of error/invalid data, display appropriate error message

# Day - 4

55. An apparel shop wants to manage the items which it sells.




**Class Description:**
**Apparel class:**

1. Initialize static variable counter to 100
2. In the constructor, auto-generate item_id starting from 101 prefixed by "C" for cotton apparels and "S" for silk apparels. Example – C101, S102, S103, C104 etc.
3. **calculate_price():** Add 5% service tax on the price of the apparel and update attribute, price with the new value

**Cotton class:**

1. While invoking parent constructor from child constructor, pass "Cotton" as item_type

2. **calculate_price():** Update attribute, price of Apparel class based on rules given below
    1. Add service tax on price by invoking appropriate method of Apparel class
    2. Apply discount on price
    3. Add 5% VAT on final price

**Silk class:**

1. While invoking parent constructor from child constructor, pass "Silk" as item_type
2. **calculate_price():** Update attribute, price of Apparel class based on rules given below
    1. Add service tax on price by invoking appropriate method of Apparel class
    2. Identify points earned based on rules given below:
    3. Silk apparels with price more than Rs. 10000, earn 10 points and anything less than or equal to that earn 3 points
    4. Initialize attribute, points with the identified points
    5. Add 10% VAT on price

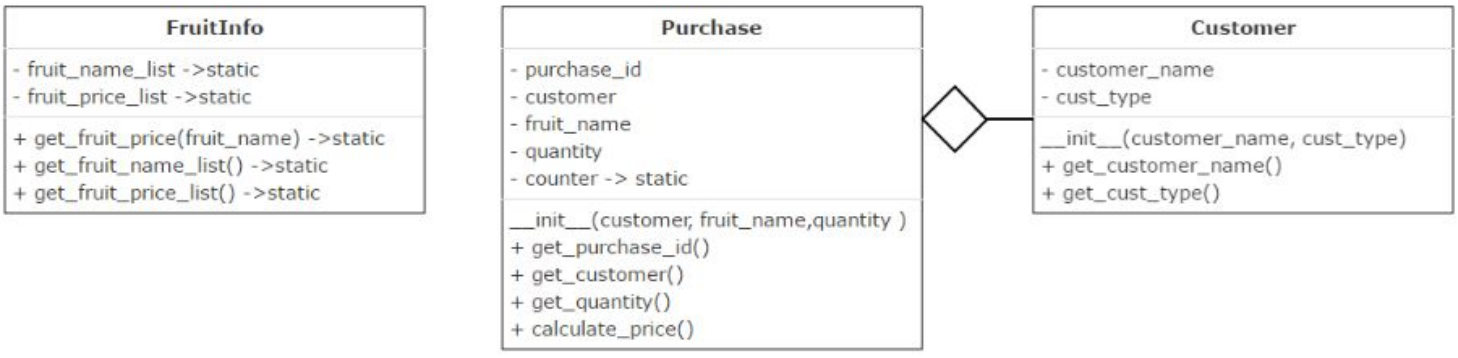
**Note:** Perform case sensitive string comparison

For testing:

- Create objects of Cotton class and Silk class
- Invoke **calculate_price()** on Cotton objects and Silk objects
- Display their details

56. **Coorg Fruit Farm** is a retail chain which sells fruits grown in their orchards in Coorg, India. They want to keep track of customers who buy fruits from them and also the billing process.

Write a python program to implement the class diagram given below.




**Class Description:**
**Fruit Info class:**

1. **fruit_name_list:** Static list which contains the list of fruits available
2. **fruit_price_list:** Static list which contains the price/kg of fruits
3. The above two lists have one-to-one correspondence, initialize it with the data given in the table
4. **get_fruit_price(fruit_name):** Accept a fruit name and return its price/kg. If fruit is not available, return -1

| **Fruit Name** | Apple | Guava | Orange | Grape | Sweet Lime |
|---|---|---|---|---|---|
| **Price per Kg** | 200 | 80 | 70 | 110 | 60 |

**Purchase class:**

1. Initialize static variable counter to 101
2. **calculate_price():** Calculate and return total fruit price based on rules given below
    1. For valid fruit name (hint: invoke get_fruit_price(fruit_name)),
        - Calculate price based on price/kg and quantity of the fruit purchased by the customer
        - If price/kg of the fruit is maximum among the fruits in fruit lists and quantity purchased is more than 1kg, apply 2% discount on calculated price
        - If price/kg of the fruit is minimum among the fruits in fruit lists and quantity purchased is 5kg or more, apply 5% discount on calculated price
        - If the customer is a "wholesale" customer, provide an additional 10% discount. Apply this discount on already discounted price, if any one of the above two points are applicable. Else apply it on calculated price
        - Auto-generate purchase id starting from 101 prefixed by “P”. Example – P101,P102 P103 etc.
        - Return final fruit price
    2. Else, return -1.

**Note:**

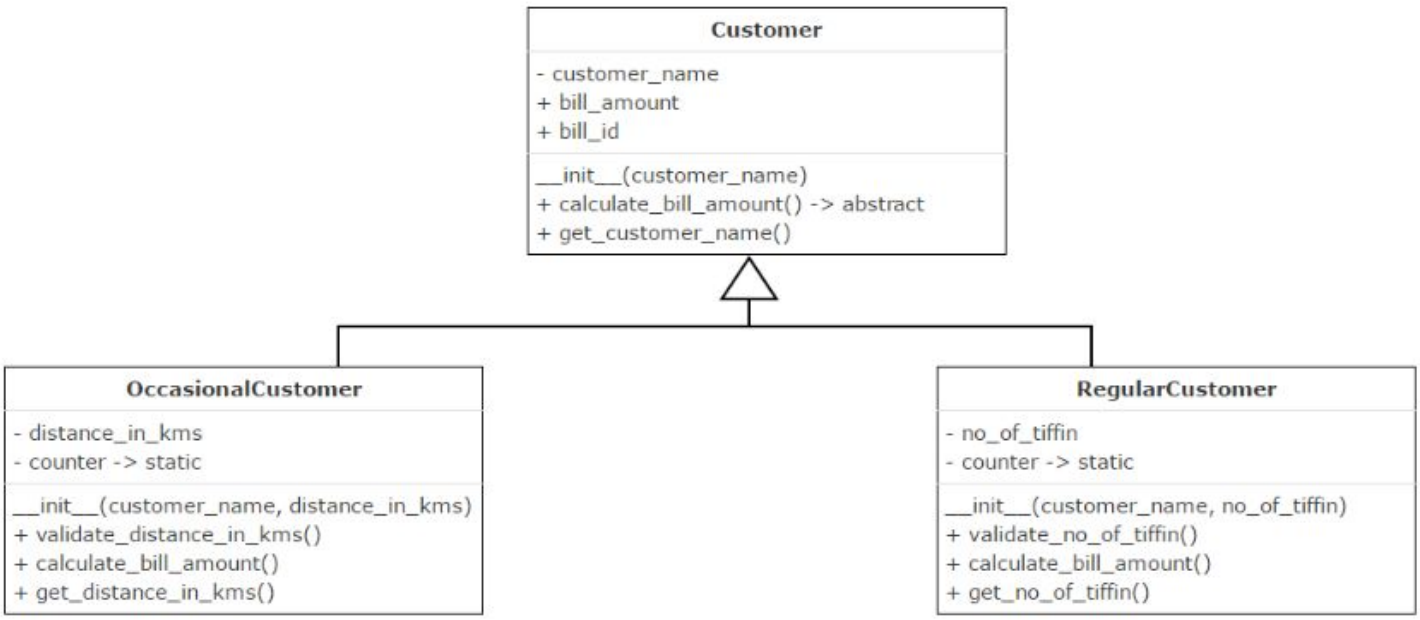
- Perform case sensitive string comparison
- There will be only one fruit with maximum price and one with minimum price

For testing:

- Create objects of Customer and Purchase class
- Invoke **calculate_price()** on Purchase object
- Display the details

# Day -5

57. Spice Hut is a tiffin service provider which home delivers dinner to their customers – Occasional customers and Regular customers.




**Class Description:**

**OccasionalCustomer class:**

1. Initialize static variable counter to 1000
2. Inside the constructor, auto-generate bill_id starting from 1001 prefixed by "O"
3. **validate_distance_in_kms():** Validate distance in kms
    1. Delivery distance in kms should be between 1 and 5 (both inclusive)
    2. If so, return true. Else, return false

4.

| **Distance in kms** | **Delivery charge in Rs.** |
|---|---|

| | |
|---|---|
| Between 1 and 2(both inclusive) | Rs. 5 per km |
| Between 2 and 5(excluding 2,including 5) | Rs. 7.5 per km |

5. **calculate_bill_amount():** Calculate total bill amount
    1. Validate distance in kms
    2. If valid, compute bill amount based on details mentioned below
        - Occasional customers can order only one tiffin per person
        - Cost/tiffin is Rs. 50
        - Delivery charges based on distance is as mentioned in the table
        - Bill amount includes tiffin cost and delivery charge
        - Set attribute, bill_amount with the computed bill amount value and return it
    3. If invalid, set attribute, bill_amount as -1 and return it

**RegularCustomer class:**

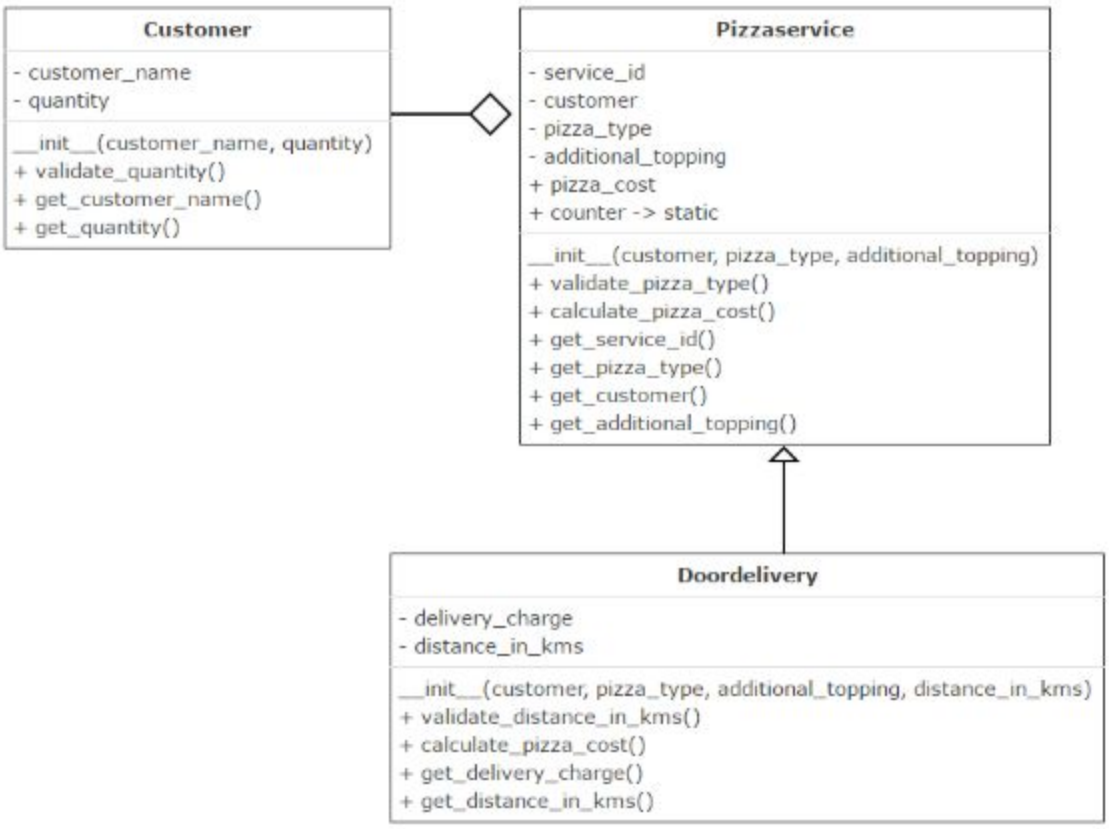
1. Initialize static variable counter to 100
2. Inside the constructor, auto-generate bill_id starting from 101 prefixed by "R"
3. **validate_no_of_tiffin():** Validate number of tiffins
    1. Regular customer can order a min of 1 and a max of 7 tiffins
    2. If value of no_of_tiffins is valid, return true. Else, return false
4. **calculate_bill_amount():** Calculate weekly bill amount
    1. Validate number of tiffins
    2. If valid, compute bill amount based on details mentioned below
        - Cost/tiffin is Rs. 50
        - The order is applicable for all the 7 days of a week
        - Compute the bill amount based on cost/tiffin, number of tiffins and number of days
        - Set attribute, bill_amount with the computed bill amount value and return it
    3. If invalid, set attribute, bill_amount as -1 and return it

**Note:** Perform case sensitive string comparison

For testing:

- Create objects of OccasionalCustomer and RegularCustomer classes
- Invoke **calculate_bill_amount()** on OccasionalCustomer and RegularCustomer objects
- Display the details of the customer

58. PizzaForYou is a pizza store which sells different kinds of pizzas based on customer's order. Customer can either collect the order in person or opt for a door delivery.
Write a python program based on the class diagram given below.




**Customer class:**

1. **validate_quantity():** A customer can order 1 to 5 pizzas
    1. If quantity is valid, return true. Else return false

**Pizzaservice class:**

1. Initialize static variable counter to 100
2. Attribute, additional_topping is a boolean value which indicates whether additional topping is required or not.
   True – additional topping is required, False – additional topping is not required
3. **validate_pizza_type():** Customers can order "small" or "medium" type pizzas
    1. If pizza type is valid, return true. Else return false
4. **calculate_pizza_cost():** Calculate pizza cost
    1. Validate pizza type and quantity
    2. If valid,

- Identify pizza cost based on details mentioned in the table
- Set attribute, pizza_cost with the calculated pizza cost
- Auto-generate service_id starting from 101 prefixed by first letter of pizza type. Example – S101, s102, m103, S104, M105 etc

3. If invalid, set pizza_cost to -1

| PizzaType | Cost/Pizza(in Rs) | Additional topping cost/Pizza (in Rs), if applicable |
|---|---|---|
| Small | 150 | 35 |
| Medium | 200 | 50 |

Doordelivery class:

1. **validate_distance_in_kms():** Minimum distance for door delivery is 1km and maximum is 10kms
    1. Validate distance_in_kms
    2. If valid, return true. Else, return false
2. **calculate_pizza_cost:** Calculate pizza cost
    1. Validate distance in kms
    2. If valid,
        - Calculate pizza cost (Hint: Invoke overridden method in parent class)
        - If pizza_cost is not -1, identify delivery charge based on details mentioned below and add it to attribute, pizza_cost

| Distance in kms | Delivery Charge in km(in Rs) |
|---|---|

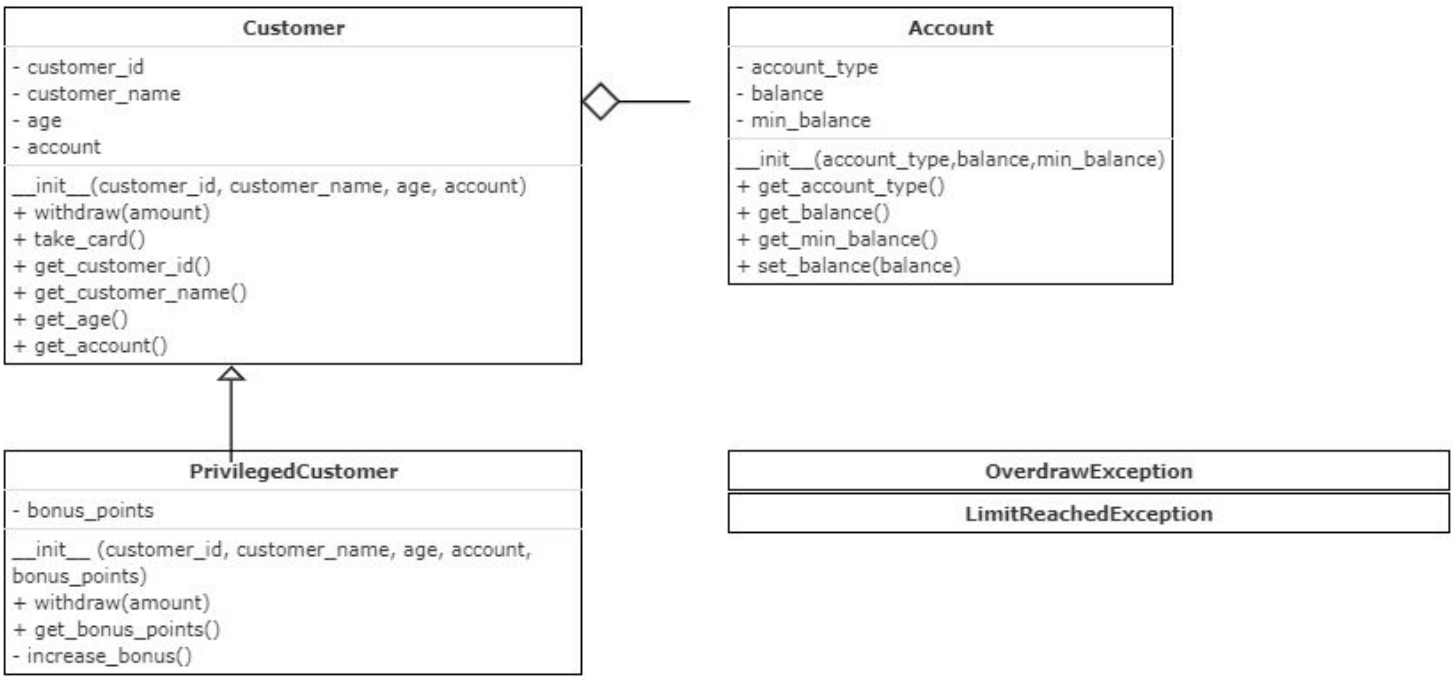
| | |
|---|---|
| For first 5 kms | 5 |
| For remaining 5 kms | 7 |

3. Else, set pizza_cost to -1

**Note:** Perform case insensitive string comparison
**For testing:**

- Create objects of Pizzaservice and Doordelivery classes
- Invoke **calculate_pizza_cost()** on Pizzaservice and Doordelivery objects
- Display the details

59. A bank has many customers and each customer has a bank account. There are also privileged customers who can earn bonus points for each of their transaction. This scenario is depicted through the below class diagram.



**Customer:**

1. **withdraw(amount):** This method should reduce the account balance based on the amount withdrawn. Raise the following exceptions based on the given conditions.

    1. OverdrawException - If the amount to be withdrawn is greater than account balance.
    2. LimitReachedException - If the balance amount is less than minimum account balance.
2. **take_card():** Displays the message "Take card out from the ATM".

**PrivilegedCustomer:**

1. **increase_bonus():** If the account balance is greater than 1000, increase the bonus points by 10, else increase it by 2.
2. **withdraw(amount):** Invoke the parent class withdraw() method and increase the bonus points by calling increase_bonus() method, if no exceptions occured.

If exceptions occur, display relevant messages. Ensure that the card is taken out from the ATM under any situation.

Write a Python program to create a new PrivilegedCustomer with below details:

Customer Id: 100

Customer Name: Gopal

Age: 43

Bonus Points: 100

Account Type: Savings

Account Balance: 1000

Account minimum: 500

The customer should be able to withdraw money and also display the bonus points of the customer after the withdraw.

## Data Structures and Algorithms

# Day 1

60. Given two lists, both having String elements, write a python program using python lists to create a new string as per the rule given below:

The first element in list1 should be merged with last element in list2, second element in list1 should be merged with second last element in list2 and so on. If an element in list1/list2 is None, then the corresponding element in the other list should be kept as it is in the merged list.

| **Sample Input** | **Expected Output** |
|---|---|
| list1=['A', 'app','a', 'd', 'ke', 'th', 'doc', 'awa']<br>list2=['y','tor','e','eps','ay',None,'le','n'] | “An apple a day keeps the doctor away” |

61. Consider a Car class as given in the code. Write a Service class as given in the class diagram below which performs various activities on a list of cars.

Assume that the car_list is sorted by year in ascending order.

| **Service** |
| --- |
| - car_list |
| __init__(car_list)<br>+ get_car_list()<br>+ find_cars_by_year(year)<br>+ add_cars(new_car_list)<br>+ remove_cars_from_karnataka() |

| **Method** | **Description** |
| --- | --- |
| __init__(car_list) | Initializes the instance variable, car_list. |
| find_cars_by_year(year) | Finds and returns the list of models of all the cars with the year as the one passed as the argument. If there are no cars, return None. |
| add_cars(new_car_list) | The new_car_list should be added to the instance variable car_list.<br>The car_list should be still sorted such that the years are in ascending order. |
| remove_cars_from_karnataka() | Finds and removes all cars with registration number beginning with “KA” from the car_list. |

62. Given two lists, both having integer elements, write a python program using python lists to create and return a new list as per the rule given below:

If the double of an element in list1 is present in list2, then add it to the new list.

| Sample Input | Expected Output |
|---|---|
| list1 - [11, 8,23,7,25, 15]<br>list2 – [6, 33, 50,31, 46, 78, 16,34] | new_list – [8,23,25] |

# Day 2

63. Given a linked list of characters. Write a python function to return a new string that is created by appending all the characters given in the linked list as per the rules given below.

**Rules:**

1. Replace '*' or '/' by a single space
2. In case of two consecutive occurrences of '*' or '/' , replace those two occurrences by a single space and convert the next character to upper case

Assume that

1. There will not be more than two consecutive occurrences of '*' or '/'
2. The linked list will always end with an alphabet

| Sample Input | Expected Output |
|---|---|
| A,n,*,/,a,p,p,l,e,*,a,/,day,*,*,k,e,e,p,s,/,*,a,/,/,d,o,c,t,o,r,*,A,w,a,y | An Apple a day Keeps A Doctor Away |

64. Write a python program to remove all duplicate elements from a sorted linked list containing integer data.

Use the LinkedList class and methods in it to implement the above program.

**Example**:

Input LinkedList: 10 10 20 20 30 30 30 40 50

Output LinkedList: 10 20 30 40 50

65. Write a python program to reverse a linked list containing integer data.

Use the LinkedList class and methods in it to implement the above program.

66. A train is identified by its name and list of compartments.

The compartments are identified by its name,total seating capacity and the number of passengers.

Implement the class Train as given in the class diagram.

| **Train** |
| --- |
| - train_name<br>- compartment_list |
| __init__(train_name,compartment_list)<br>+ get_train_name() |

+ get_compartment_list()

+ count_compartments ()

+ check_vacancy()

Write a python program to implement the following:

1. count_compartments()- returns the total number of compartments in the train
2. check_vacancy()-returns the count of the compartments in which more than 50% of the seats are vacant

**Note :** Compartment list is maintained as a linked list where data in each node refers to a compartment.

# Day 3

68. Given two queues, one integer queue and another character queue, write a python program to merge them to form a single queue.  Follow the below rules for merging:

- Merge elements at the same position starting with the integer queue.
- If one of the queues has more elements than the other, add all the additional elements at the end of the output queue.

Note : max_size of the merged queue should be the sum of the size of both the queues.

For example,

Input -- queue1: 3,6,8  queue2: b,y,u,t,r,o

Output -- 3,b,6,y,8,u,t,r,o

69. Given a stack of integers, write a python program that updates the input stack such that all occurrences of the smallest values are at the bottom of the stack, while the order of the other elements remains the same.

For example:

Input stack (top-bottom) :  5 66  5  8  7

Output:  66  8  7  5  5

70. Given a queue of whole numbers. Write a python function to return a new queue which contains the evenly divisible numbers.

**Note:** A number is said to be evenly divisible if it is divisible by all the numbers in the given range without any remainder. Consider the range to be from 1 to 10 (both inclusive).
Assume that there will always be few elements in the input queue, which are evenly divisible by the numbers in the mentioned range.

Example:
Input Queue: 13983, 10080, 7113, 2520, 2500 (front - rear)
Output Queue: 10080, 2520 (front-rear)

71. The Lenovo Laptop Service center provides different services to its customers. As and when a job is arrived, it is allocated to an employee, if no job is assigned. Otherwise it is added to

pending jobs queue. On completion of job by an employee based on elapsed time in minutes, a job in pending queue is allocated to the corresponding employee.

Assumptions:

1. The service center allows to have a maximum of 10 jobs in the pending queue.
2. An employee is allocated only one job at a time
3. Each job will take a maximum of 1 hr to complete, time needed for completing a job and time elapsed are given in minutes.

## Day - 4

86. The International Cricket Council (ICC) wanted to do some analysis of international cricket matches held in last 10 years.

Given a list containing match details as shown below:
[match_detail1,match_detail2……]

Format of each match_detail in the list is as shown below:
**country_name : championship_name : total_number_of_matches_played : number_of_matches_won**
Example: AUS:CHAM:5:2 means Australia has participated in Champions Trophy 5 times and have won 2 times.

Write a python program which performs the following:

**find_matches (country_name):** Accepts the country_name and returns the list of details of matches played by that country.

**max_wins():** Returns a dictionary containing the championship name as the key and the list of country/countries which have won the maximum number of matches in that championship as the value.

**find_winner(country1,country2):** Accepts name of two countries and returns the country name which has won more number of matches in all championships. If both have won equal number of matches, return "Tie".

Perform case sensitive string comparison wherever necessary.

match_list – ['ENG:WOR:2:0', 'AUS:CHAM:5:2', 'PAK:T20:5:1', 'AUS:WOR:2:1', 'SA:T20:5:0', 'IND:T20:5:3', 'PAK:WOR:2:0', 'SA:WOR:2:0', 'SA:CHAM:5:1', 'IND:WOR:2:1']

| Sample Input | Expected Output |
|---|---|
| find_matches ("AUS") | ['AUS:CHAM:5:2', 'AUS:WOR:2:1'] |
| max_wins() | {'WOR': ['AUS', 'IND'], 'CHAM': ['AUS'], 'T20': ['IND']} |
| find_winner("AUS","IND") | IND |

87. Mary is a kindergarten teacher. She has given a task to the children after teaching them a list of words. The task is to find the unknown words (other than the words they already know) from the given text. Write a python function which accepts the text and the known list of words and returns the set of unknown words found.

Return -1 if there are no unknown words.

| Sample Input | Expected Output |
|---|---|
| Text: "the sun rises in the east"<br>Vocabulary: ["sun","in","east","doctor","day"] | {'rises', 'the'} |

88. Given a list of numbers sorted in ascending order. Write a python function which searches for a given number in the list. The given number may occur more than once in the list. The function should return the index position at which the last occurrence of the given element is found. If the number is not found, return -1.

# Day 5

89. The central library at Mysore has a set of very interesting books and journals. The books are arranged in the alphabetical order of their author names. So it is very easy for the readers to search for the book.

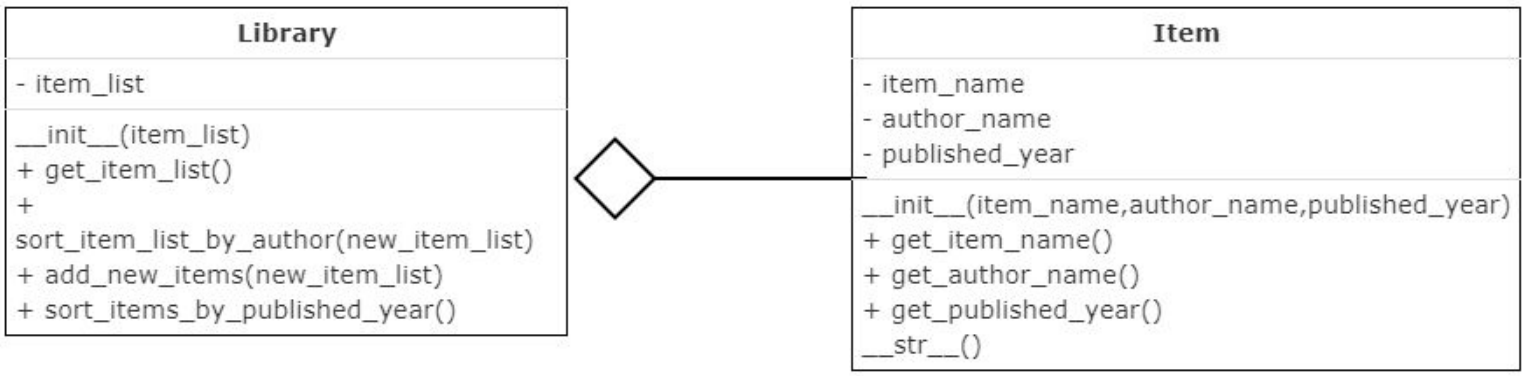
The library has got a set of new books. The librarian wants to arrange them in order too. As some books are already arranged in the order, find a suitable way of arranging the new set of books amidst them.

Write a python program to arrange all the books in the alphabetical order of the author names.
**sort_item_list_by_author():** Accepts the new list of books and returns it sorted in the alphabetical order of their author names.

**add_new_items():** Accepts the new list of books, sorts it and merges it with the existing books in the library.
**Hint** - Use sort_item_list_by_author() method for sorting the books.

**sort_items_by_published_year():** Sorts the list of books (item_list) based on the increasing order of their published years. If there are multiple items that are published in the same year, then sort them based on the alphabetical order of their author names.

**Note:** While sorting the author names in alphabetical order, ignore the special characters including space, if there are any.




90. The manager of an airport wants to generate various reports.

Details of the flights and passengers are stored as mentioned below:

1. Flight details are stored as a list of strings. Suppose "AI890:BAN:MUM:1400" is a string in the flight details list, it should be interpreted as follows:
   AI890 is the flight name , BAN is the source, MUM is the destination and 1400 is the departure time (24 hour format).
2. Passenger details are stored in a dictionary where key is the PNR number of the passenger and value is a list containing passenger details. Suppose "LW101":["Amanda","AI678","C7",25] is an element in the dictionary, it should be interpreted as follows:
   LW101 is the PNR number of the passenger and ["Amanda","AI678","C7",25] is the list in

which the index 0, 1, 2 and 3 represents the passenger name, flight name, seat number and the baggage weight respectively.

Assume that we are considering only those flights which are departing between 0700hrs and 2000hrs.

Write a python program to perform the below mentioned functionalities.

**find_flights(flight_time):** This method accepts time in 24 hour format and returns the list of flights which are waiting to takeoff within another two hours starting from the given time (both inclusive).

**sort_flight_list(flight_list):** This method takes the flight details list as the input and returns the flight details list sorted based on the increasing order of their departure time.

**get_passenger_details(flight_detail):** This method takes a flight's detail as input and returns the list of PNR numbers of the passengers who are waiting to board the given flight.

**security_check(passenger_pnr_list):** This method takes the list of PNR numbers of the passengers boarding a specific flight as the input and returns the list of PNR numbers of the passengers whose baggage has been cleared.

The baggage will be cleared if the baggage weight is between 0-25kg (both inclusive)

**sort_passengers(passenger_pnr_list):** This method takes the list of PNR numbers of the passengers whose baggage has been cleared as the input and returns the list of PNR numbers sorted based on the increasing order of their seat numbers. ( order of seats: A→J )

**boarding(passenger_pnr_list):** The passengers who have to board a flight should stand in a queue. This method takes the list of PNR numbers of the passengers sorted based on seat numbers as the input and returns the queue of PNR numbers of the passengers. The queue should be of the same size as that of the list. The first passenger in the list should be the first person to stand in the queue.

**seating(passenger_queue):**The flight has only one door which is at the back side. The passengers who board the flight are expected to occupy seats from the front. So the passenger who board last will be able to come out first from the flight. This method takes the queue of PNR numbers of the passengers as the input and returns the stack ( max size of the stack should be same as the size of the queue) which contains the PNR numbers of the passengers representing the seating.

91. Write a python function which accepts two sorted stacks and returns a new stack containing all the elements of input stacks in sorted order.

**Note:** The output stack should be of the size as that of the sum of the sizes of the input stacks.

| Sample Input | Expected Output |
| --- | --- |

| | |
|---|---|
| Stack1 (top - bottom): 7,6,5<br>Stack2 (top - bottom): 3,2,1 | Stack (top-bottom) : 7,6,5,3,2,1 |
| Stack1 (top - bottom): 15,10,3<br>Stack2 (top - bottom): 21,9,7 | Stack (top-bottom) : 21,15,10,9,7,3 |

**Day 6**

92. Given a list of digits. The task is to find out the number of possible decoding of the given digit sequence.

Assume that the input list contains valid digits from 0 to 9 and there are no leading 0's, no trailing 0's and no two or more consecutive 0's in the input list.

Let 1 represent 'A', 2 represent 'B', …, 26 represent 'Z'.

Write a python function which accepts the list of digits and returns the total number of decoding possible from given digit sequence.

| Sample Input | Expected Output | Remark |
|---|---|---|
| digit_list=[1 ,2 ,2 ,4] | 5 | The possible decoding are "ABBD", "LBD", "AVD", "ABX","LX" |

| | | |
|---|---|---|
| digit_list=[1 ,2 ,2] | 3 | The possible decoding are "ABB", "LB", "AV" |

93. The Bangalore railway authorities have decided to construct the required number of platforms in the station so that the trains need not wait in order to get the platforms.

Given the arrival and departure times of all the trains that reach a railway station. The task is to find the minimum number of platforms required for the railway station so that no train waits without the platform.

Write a python function which accepts the arrival time list and departure time list and returns the minimum number of platforms required.

| Sample Input | Expected Output |
|---|---|
| arrival_time_list = [800,850,600,1120,1050,900]<br>departure_time_list = [1110,1200,1400,1130,1700,2200] | 5 |
| arrival_time_list = [800,850,600, 1350, 1120,1050,900]<br>departure_time_list = [1110,1200,830, 1400, 1055, 1130,1700,2200] | 4 |

Note: Time is provided in 24 hour format.

Hint: Start by sorting the arrival time list and departure time list and then compare the arrival and departure time.

94. Write a python function which accepts a given number, n and returns the count of all possible distinct binary strings which are of length n, such that there are no consecutive 1's in the binary string.

| Sample Input | Expected Output |
|---|---|
| n=2 | 3 |
| n=3 | 5 |

**Note:** The possible binary strings when n= 2 are 00, 01 and 10 and the possible strings when n=3 are 000,001,010,100,101.

95. Given a queue of numbers. Write a python function to push every second element in the queue to a stack, if it is the triangle number of the previous element in the queue and return the stack. The output stack should be of the same size as that of the input queue.

Number d is said to be a triangle number of n, if d = 1 + 2 + 3 +….+ (n-2) + (n-1) + n.

For example, 28 is said to be the triangle number of 7, if 1+2+3+4+5+6+7 is equal to 28.

| Sample Input | Expected Output |
| --- | --- |
| Input queue (front->rear): 7,28,8,35,3,6,5,15,2,5 | Output stack ( top->bottom): 15,6,28 |

96. Given a stack with strings, A through E, in alphabetical order with "A" on top.

Write a python function to accept the given stack and perform the following:
The bottom three elements should be simultaneously removed and placed on top with their vertical order maintained and it should return the updated input stack.

Input stack (top->bottom): A,B,C,D,E

Output stack (top->bottom) :C,D,E,A,B

97. Write a python function to implement separate chaining technique for handling collision in a hash table. The function should accept the list of whole numbers that need to be hashed and the size of the hash table, n. It should store the whole numbers in the linked list at the appropriate index positions of the hash table and return the hash table.

Assume that the hash function is h(k)=k % n, where k is the number to hashed and n is the size of the hash table.

Hint: Consider each element of the hash table to be a linked list.

Example: hash_table[0]=LinkedList(), hash_table[1]=LinkedList() etc.