SHORT PROGRAMS

Define a class named **NUMBER** consisting of the following:

1. i) **num**: of type short unsigned integer under private visibility label.
   ii) Define a private member function named "**input**" to accept the data member "num", determine and return 1 if the "num" is palindrome else it return 0.
   iii) Define a public member function named "**display**" which call the member function 'input" and display whether the number is palindrome or not.

   Write a relevant main function to complete the program.

   NOTE:- A positive integer number is a "Palindrome Number" if its reversal is equal to original number.

2. Define a class named **FIBO** consisting of the following members:
   i) **N**: of type unsigned short integer under the private visibility label.
   ii) Define a private member function named "**input**" to accept value 'N' from the user.
   iii) Define a public member function named "**display**" which call the member function "input" and display the Fibonacci series up to the 'N' terms.

   Write a relevant main function to complete the program.

   NOTE:- Fibonacci number is an integer in the infinite series 0,1,1,2,3,5,8……… of which the first two terms are 0 and 1 and each succeeding term is the sum of the previous two terms.

3. Define a class named **PRIME** consisting of the following members:-
   i) **num**:- of type unsigned short integer under private visibility label
   ii) Define a private member function named **get_no( )** to accept a positive number from the user.
   iii) Define a public member function named **process()** which call the member function get_no() ,checks and displays if num is prime or composite number.

   Write a relevant main function to complete the program .

   NOTE:- a positive intger number is said to be prime if it is only divisible by 1 and itself.
   Egs of prime numbers:- 2, 3, 5, 7, 11 ……..
   Egs of Composite numbers:- 4, 6, 9, …..
   Number 1 is neither prime nor composite.

4. Define a class named **ARMSTRONG** consisting of the following members:
   i) **num**: of type short unsigned integer under private visibility label.
   ii) Define a public member function "**get_no**" to read a three digit positive integer no from the user and call the member function "process"
   iii) Define a private member function "**process**" which checks and display if num is a Armstrong number or not.

   Write a relevant main function to complete the program

Note:A three digit positive integer number is a "Armstrong Number " if sum of cubes of its digits is equal to the given number itself.
Example 153 is a Armstrong number .
1*1*1+5*5*5+3*3*3=1+125+27=153

5. Define a class named **SERIES** consisting of the following Members:
   i) **N:** of type unsigned short integer under private visibility label. (N indicates number of terms)
   ii) **D**: of type float under private visibility label. (D indicates angle in degree)
   iii) Define a public member function "**get_data**" to read values for the data members D and N.
   iv) Define a private member function "**process**" to compute sin(x) by determining the summation of first "N' terms of the following series for display.

$\cos(x)= \frac{x}{1!} - \frac{x3}{3!} + \frac{x5}{5!} + \frac{x7}{7!} + \frac{x^9}{9!} + \ldots\ldots\ldots\ldots\ldots + \frac{x^N}{N!}$ where x is angle in radians.

Write a relevant main function to complete the program.

6. Define a class named **SMALLER** consisting of the following members:-
   i) d1, d2, d3 of type double under private visibility label
   ii) Define a private member function named " process" which finds the smaller of d1, d2, and d3 and displays the smallest
   iii) Define a parametrized constructor function which initialises the private data members.

   Write the relevant main function to complete the program.

7. Define a class **BASE** having one private data member num1 and one public data member **num2** both of type float. Define public member functions:
   i) **input_data()-** to read data value num1.
   ii) **get_num1()-** to return the value of num1.
   Extend class **BASE** to another class **DERIVED** using public derivation. Define for class DERIVED a private data member sum which is to be calculated by adding num1 and num2 and a public member function:
   get_data()- to read num2 and to call input_data() for reading value to add to compute sum.
   show_data()-to output num1, num2 ans sum.
   Write a main() to create object of type DERIVED and input and output all data.

8. Define an abstract class named "**BASE1**" which has the following members:
   i) **A** : an integer type variable under private visibility label
   ii) Define an inline parameterized constructor to initialize data member "A".
   iii) **Get_A()**: an inline protected member function which returns value of "A".
   Define an abstract class named "**BASE2**" which has the following members:
   i) B: an integer type variable under private visibility label
   ii) Define a inline parameterized constructor to initialize data member "B".
   iii) Get_B: an inline public member function which returns value of "B".
   Define a class named "**DERIVED**" which is derived from "BASE1" and "BASE2" under

public and **protected** mode respectively. It has the following members:

i) Z: an integer type variable under private visibility label.

ii) show(): an inline private member function which displays value of Z.

iii) Define an inline parameterized constructor to initialize data member "Z" with product of "A" and "B", it further calls member function show().

Define an appropriate main function.

9. Define a class named "**TIME**" which has the following members:
   - **Hours, Minutes**: of type integer under private visibility label
   - Define a **parameterized constructor** to initialize data members "Hours" and "Minutes.
   - **Sum()**: a public member function which accepts two objects as parameter of type "TIME". It calculates summation of two time quantities represented by parameters and assigns it to data members of object which has called member function sum().
   - **Display()**: a public member function to display data members of object of type "TIME".

   Define an appropriate main() function to display addition of two objects of type "TIME".

10. Define a class named **VOLUME** consisting of the following members:

    i) **r,r1,h,v**: of type float under private visibility label.

    ii) Define a **parameterized constructor** to initialize data member "r"

    iii) Define a **parameterized constructor** to initialize data members "r1" and "h".

    iv) **Display()**: to display Volume of sphere ($v=4/3\pi r^3$) and volume of cylinder ($v=\pi r^2 h$)

    Write the relevant main function to complete the program.

LONG PROGRAMS

1 Define a class named **BINARY** consisting of the following members:

i) **list**: an array of type short integer of size 30 under private visibility label.

ii) **N** : of type short unsigned integer (indicates total number of elements to be accepted in array "list") under private visibility label.

iii) Define a **default constructor** to accept data member "N" and to accept the numbers in the array "list". It further calls member function "search".

iv) Define a private member function named "**search**" which accepts the number to be searched from the user and determines whether it is present in the "list" using **binary search technique**.

Write a relevant main function to complete the program.

2. Define a class named **BUBBLE** consisting of the following members:

i) **list**: an array of type short integer of size 30 under private visibility label.

ii) **N** : of type short unsigned integer(indicates total number of elements to be accepted in array "list")under private visibility label.

iii) Define a **default constructor** to accept data member "N" and to accept the numbers in the array "list" .It further calls member function "sort" followed by member function "show".

iv) Define a private member function named "**sort**" which performs sorting of numbers in array "list" using **bubble sort technique**(sort in **ASCENDING** order).

v) Define a private member function named "**show**" which displays the content of array.

Write a relevant main function to complete the Program.

3. Define a class named **SELECT** consisting of the following members:

i) **list:** an array of type short integer of size 30 under private.

ii) **N** :of type short unsigned integer.(indicates total number of elements to be accepted in array "list") under private .

iii) Define a **default constructor** to accept data member "N" and to accept the numbers in the array "list". It further calls member function "sort" followed by member function "show".

iv) Define a private member function named "**sort**" which performs sorting of numbers in the array "list" using **selection sort technique** (sort in **ASCENDING** order).

v) Define a private member function named "**show**" which displays the content of array.

Write a relevant main function to complete the program.

4. Define a class named **INSERT** consisting of the following members:
   i) **list**: an array of type short integer of size 30 under private visibility label.
   ii) **N** : of type short unsigned integer(indicates total number of elements to be accepted in array "list") under private visibility label.
   iii) Define a **default constructo**r to accept data member "N" and to accept the numbers in the array "list". It further calls member function "sort" followed by member function "show".
   iv) Define a private member function named "**sort**" which performs sorting of numbers in the array "list" using **insertion sort** technique (sort in **ASCENDING** order).
   v) Define a private member function named "**show**" which displays the content of array.

   Write a relevant main function to complete the program.

5. Define a class named **MERGE** consisting of the following members:
   i) **A,B,C**: 1-D arrays of type short integer of size 50 each under private visibility label.
   ii) **M,N** : of type short unsigned integer (indicates total number of elements in A and B respectively) under private visibility label.
   iii) Define a **default constructor** to accept data members "M" and "N" and to accept the numbers in the arrays A and B (both in **ascending** order). It further calls member function "process".
   iv) Define a private member function named "**process**" which performs **merging** of all elements in A and B to obtain array C in **ASCENDING** order for display.

   Write a relevant main function to complete the program.

6. Define a class named **MATRIX** consisting of the following members:
   i) **M.N,Q**: of type short unsigned integer under private visibility label.
   ii) **A,B,C** : 2-D arrays of size 10x10 each under private visibility label.
   iii) Define a **default constructor** to accept data members M,N and Q (Where MxN is size of A and NxQ is size of B). It also accepts 2-D array A and B.

      It further calls member function "product".
   iv) Define a private member function named "**product**" which computes product of two matrix's A and B and stores in C. It further displays matrix C in tabular form.

Write a relevant main function to complete the program.

7. Write a menu driven program to implement a **singly linked list** in which each node consists of the following data fields:
   i) **M.N,Q**: of type short unsigned integer under private visibility label.
   ii) **roll**- of type unsigned short integer.
   iii) **name**- an array of maximum 30 characters.
   iv) **percent**- of type float

   And perform the following operations:

   a) **Creation** of a linear linked list containing "n" nodes.
   b) **Display** the linear linked list in the following format:

ROLL	NAME	PERCENT
1000	SACHIN	89.9
2000	KARUN	75.52
3000	KIERA	65.18

8. Write a menu driven program to implement a **stack** using singly linked list in which each node consists of a single data field of type **character** and performs the following operations:
   i) **push** a node onto the stack
   ii) **pop** the top node from the stack

   Displaying data field of all the nodes in the stack horizontally.

9. Write a menu driven program to implement a **queue** using singly linked list in which each node consists of a single data field of type character and performs the following operations:
   i) **Appending** a node into the queue.
   ii) **Deleting** the first node in the queue.

   Displaying data field of all the nodes in the queue horizontally.

10. Define a class named **BILL** with the following members:
    i) **item_code**: of type unsigned short integer under private visibility label.
    ii)**item _name**: a character array of size 30 under private visibility label.
    iii) **unit_price,total** : of type float under private visibility label.
    iv) **quantity**: of type unsigned short integer under private visibility label.
    v) Define a member function named "**get data**" to accept data members item_code, item_name,unit_price and quantity. It computes total as quantity * unit_price.
    vi) Define a member function named "**put data**" to display data members item_code, item_name,unit _price,quantity and total.

Write a menu driven main function to

a) **Create** a binary file named “market.data” containing objects of type BILL.

b) **Display** all the data members of the objects read from file “market.data” in tabular form.

<u>**SAMPLE OUTPUT**</u>

```
ITEM CODE    ITEM NAME   UNIT-PRICE      QUANTITY   TOTAL
  1000          PEN         10.00            5          50.00
  2000        PENCIL         5.00            3          15.00
```