Q1. Write a program by creating a class Bicycle as a base class with a number of gears and speed of bicycle as integer attributes and create a class called MountainBike, a derived class that extends Bicycle class with an attribute seat height as an integer. Create a Test class to run the program and obtain the output in the console.
Note: Override toString() method to display the details of the bicycle.

Input Format

To get 3 integers from the user (Number of gears, Speed of bicycle, and Seat height).

Output Format

To display the desired output from the test class.

Constraints

integers only.

Sample Input

```
2 90 40
```

Sample Output

```
No of gears are 2 speed of bicycle is 90 seat height is 40
```

Sample Input

```
3 60 20
```

Sample Output

```
No of gears are 3 speed of bicycle is 60 seat height is 20
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. Write a program by creating a class Bicycle as a base class with a number of gears and speed of bicycle as integer attributes and create a class called MountainBike, a derived class that extends Bicycle class with an attribute seat height as an integer. Create a Test class to run the program and obtain the output in the console.
Note: Override toString() method to display the details of the bicycle.

Input Format

To get 3 integers from the user (Number of gears, Speed of bicycle, and Seat height).

Output Format

To display the desired output from the test class.

Constraints

integers only.

Sample Input

```
2 90 40
```

Sample Output

```
No of gears are 2 speed of bicycle is 90 seat height is 40
```

Sample Input

```
3 60 20
```

Sample Output

```
No of gears are 3 speed of bicycle is 60 seat height is 20
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q3. Develop a program for the banking system for account management. Each account has the following attributes: **AccountID, HolderName, and Balance**. Declare one constructor with three parameters that initialize the three attributes to some default values. Attributes must be validated.

- **AccountBalance** must be greater than or equal to zero. If not, it is set to zero.
- **AccountID** must be between 100 and 999. If not, set it to -1 to indicate that it is invalid.

Use the method **setAccountBalance (…)** to print the account balance. Write one method **Credit** to deposit money into the account. The method should return the new balance after money deposit. Then create a class **VIPAccount** that inherits from the class Account. The **VIPAccount** class overrides the method **setAccountBalance (…)** such that it prints the balance can be negative but no less than **– 10000.** The constructor of the VIPAccount class must call the constructor of the Account class.

Input Format

The first line of the input consists of the account id. The next input is the account holder's name.
The third input is the initial balance.
The fourth input is the amount to be credited.
The last input is a negative balance (Argument to setAccountBalance in overridden method).

Output Format

The first line of the output prints the account details.
The next line prints the new balance after the amount is credited.
The next output is the result of setAccountBalance (First base class method then derived class method).

Sample Input

```
120
Alice 48200
```

Sample Output

```
120 Alice 48200 48700
48700
```

Sample Input

```
10
Bob 120
```

Sample Output

```
-1 Bob 120
220
220
```

Sample Input

```
848
Charlie
-120
```

Sample Output

```
848 Charlie 0
52040
52040
```

Sample Input

```
1288
David 48484
```

Sample Output

```
-1 David 48484
133332
133332
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q4. Create an abstract class **Shape** with length, width, radius, 3 sides as data members and two abstract methods to calculate area and perimeter. Create constructors and getter setters.

Create four classes **Square**, **Rectangle**, **Circle** and **Triangle.** Extend all the classes from **Shape** directly**.** Complete the abstract method to calculate area and perimeter in the derived classes.

Get a single character and suitable values from user to calculate area and perimeter.

Input Format

S or R or C or T in first line (S represents Square, R represents Rectangle, C represents Circle and T represents Triangle) Enter one or two input based on Shape (1 input for Square and Circle, 2 inputs for Rectangle and 3 inputs for Triangle)

Output Format

Perimeter or Circumference

Area

Sample Input

```
S 5
```

Sample Output

```
Perimeter : 20.00
Area : 25.00
```

Sample Input

```
R 3
4
```

Sample Output

```
Perimeter : 14.00
Area : 12.00
```

Sample Input

```
C 7
```

Sample Output

```
Circumference : 43.98
Area : 153.94
```

Sample Input

```
T 3
4
```

Sample Output

```
Perimeter : 12.00
Area : 6.00
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. Write a program to implement the following logic using inheritance.
Create a parent class and implement the fun method. In the method, get the individual digits of the entered number, store it in an array, and find their sum. For example in case of 1234, the individual digits are 4,3,2,1 and the final sum → (4+3)+(4+2)+(4+1)+
(3+2)+(3+1)+(2+1) = 30. Create the main class that inherits the parent class and call the fun method inside the parent function.

Input Format

The input consists of an integer.

Output Format

The output prints the final sum.

Constraints

Integers only.

Sample Input

```
1234
```

Sample Output

```
30
```

Sample Input

```
4356
```

Sample Output

```
54
```

Memory Limit: - kb Code Size: - kb