1. Create a **Person** class with private fields name and age, and public getter and setter methods for each field. Then, create an instance of the Person class and set the values of the fields using the setter methods. Finally, print the values of the fields using the getter methods.

2. Create a BankAccount class with private fields accountNumber, balance, and interestRate, and public getter and setter methods for each field. The balance field should be of type double, while the other two fields should be of type int. Add a public method called deposit that takes a double parameter and adds it to the balance. Add a public method called withdraw that takes a double parameter and subtracts it from the balance. Make sure that the withdraw method checks that the balance is sufficient to cover the withdrawal amount, and throws an IllegalArgumentException if the balance is insufficient.

3. Create a Vehicle class with a private field maxSpeed and a public method startEngine that prints "Engine started". Then, create a Car class that extends Vehicle and adds a private field numberOfDoors. Add public getter and setter methods for numberOfDoors, and override the startEngine method to print "Car engine started". Finally, create an instance of Car and call the startEngine method to verify that it prints "Car engine started".

4. Create a Shape class with a private field color and a public method getColor that returns the value of color. Then, create a

Circle class that extends Shape and adds a private field radius. Add public getter and setter methods for radius, and override the getColor method to print the color of the circle in addition to returning it. Finally, create an instance of Circle and call the getColor method to verify that it prints the color of the circle.

Practice Arrays 1

---

1. --Create an array of 10 integers and initialize each element to 0.
2. Create an array of 5 doubles and initialize each element to 1.0.
3. --Create an array of 7 strings and initialize each element to an empty string.
4. Declare an array of 100 integers and fill it with random numbers between 1 and 1000.
5. --Create an array of 4 integers and find the sum of all the elements.
6. Create an array of 6 doubles and find the average of all the elements.

7. --Create an array of 8 strings and print out the length of each string.

8. Create an array of 10 integers and find the largest element.

9. Create an array of 12 doubles and find the smallest element.

10. Create an array of 15 integers and sort the elements in ascending order.

11. Create two arrays of 5 integers each and add the corresponding elements of the arrays.

12. Create two arrays of 7 doubles each and multiply the corresponding elements of the arrays.

13. Create an array of 20 integers and remove all duplicates from the array.

14. Create an array of 25 integers and find the first occurrence of the number 10.

15. Create an array of 30 strings and find the index of the first occurrence of the string "hello".

Practice Arrays 2

---

1. Create a class called Person that has private fields for name, age, and address. Include getter and setter methods for each field. Create an array of 5 Person objects and print out the name and age of each person.

2. Create a class called Car that has private fields for make, model, and year. Include getter and setter methods for each field. Create an array of 10 Car objects and print out the make, model, and year of each car.

3. Create a class called BankAccount that has private fields for account number, balance, and owner name. Include getter and setter methods for each field. Create an array of 3 BankAccount objects and print out the account number, balance, and owner name of each account.

4. Create a class called Book that has private fields for title, author, and number of pages. Include getter and setter methods for each field. Create an array of 7 Book objects and print out the title and author of each book.

5. Create a class called Student that has private fields for name, age, and grades (an array of integers). Include getter and setter methods for name and age, and a method to calculate the average grade. Create an array of 4 Student objects and print out the name, age, and average grade of each student.

6. Create a class called Rectangle that has private fields for length and width. Include getter and setter methods for each field, and a method to calculate the area of the rectangle.

Create an array of 6 Rectangle objects and print out the length, width, and area of each rectangle.

7. Create a class called Employee that has private fields for name, salary, and department. Include getter and setter methods for each field. Create an array of 5 Employee objects and print out the name, salary, and department of each employee.

8. Create a class called Circle that has a private field for radius. Include getter and setter methods for the radius, and a method to calculate the circumference of the circle. Create an array of 9 Circle objects and print out the radius and circumference of each circle.

9. Create a class called Animal that has private fields for name and species. Include getter and setter methods for each field. Create an array of 8 Animal objects and print out the name and species of each animal.

10. Create a class called Course that has private fields for name and students (an array of Student objects). Include getter and setter methods for the name and students, and a method to calculate the average grade of all the students in the course. Create an array of 2 Course objects, each with 3 students, and print out the name of each course and the average grade of all the students in the course.

---

practice about using object as arguments of methods or return type of methods

---

1. Write a method that takes in an array of Person objects and returns the average age of those people.

2. Create a class called Circle with a radius property. Write a method that takes in an array of Circle objects and returns the total area of all the circles in the array.

3. Write a method that takes in an array of Book objects and a string representing a category (such as "mystery" or "romance"). The method should return an array of only the books in the specified category.

4. Create a class called Student with properties for name and GPA. Write a method that takes in an array of Student objects and returns the name of the student with the highest GPA.

5. Write a method that takes in an array of Car objects and a string representing a make (such as "Toyota" or "Ford"). The method should return an array of only the cars made by the specified make.

6. Create a class called BankAccount with properties for balance and account number. Write a method that takes in an array of

BankAccount objects and returns the total balance of all the accounts in the array.

7. Write a method that takes in an array of Employee objects and a double representing a bonus percentage. The method should add the bonus to each employee's salary and return the total amount of the bonuses given.

8. Create a class called Product with properties for name and price. Write a method that takes in an array of Product objects and returns the name of the product with the lowest price.

9. Write a method that takes in an array of Rectangle objects and returns the total perimeter of all the rectangles in the array.

10. Create a class called Customer with properties for name and total purchases. Write a method that takes in an array of Customer objects and returns the name of the customer with the highest total purchases.

---

practice

---

1. Create a class Animal with a String field name and an int field age. Create a constructor that initializes these fields. Create a method speak that prints "I am an animal."

2. Create a class Dog that extends Animal. Add a String field breed and a constructor that initializes it. Override the speak method to print "I am a dog."

3. Create a class Cat that extends Animal. Add a boolean field isIndoor and a constructor that initializes it. Override the speak method to print "I am a cat."

4. Create a class Bird that extends Animal. Add a String field species and a constructor that initializes it. Override the speak method to print "I am a bird."

5. Create a class Zoo with a List of Animal objects. Add methods to add animals to the list, remove animals from the list, and print all animals in the list using their speak method. Use the super keyword to call the constructor of the parent class in each of the constructors of the child classes.

6. Create a class Fish that extends Animal. Add a String field waterType and a constructor that initializes it. Override the speak method to print "I am a fish."

7. Modify the Zoo class to handle Fish objects in addition to Animal objects. Add methods to add fish to the list, remove fish from the list, and print all fish in the list using their speak method.

8. Create a class Mammal that extends Animal. Add a boolean field isWarmBlooded and a constructor that initializes it. Override the speak method to print "I am a mammal."

9. Create a class Human that extends Mammal. Add a String field firstName, a String field lastName, and a constructor that initializes these fields. Override the speak method to print "My name is [first name] [last name]."

10. Create a class Student that extends Human. Add an int field gradeLevel and a constructor that initializes it. Override the speak method to print "I am a student in grade [grade level]." Use the super keyword to call the constructor of the parent class in each of the constructors of the child classes.

Ternary Operator:

1. Write a Java program that takes an integer as input from the user and determines whether it is positive, negative, or zero. Use the ternary operator to accomplish this task.

2. Write a Java program that takes three integer values as input from the user and determines which one is the largest. Use the ternary operator to accomplish this task.

3. Write a Java program that takes a string as input from the user and determines whether it contains the substring "hello". If it does contain the substring, the program should print "The string contains hello", otherwise it should print "The string does not contain hello". Use the ternary operator to accomplish this task.