Assignment1:

For the given Object solve the following:

```
var people = [
    {
          "firstname": "praveen",
          "lastname": "gubbala",
          "age": 36,
          "gender": "male",
           "city": "hyd",
          "salary": 10000
   },
   {
          "firstname": "srikanth",
          "lastname": "gubbala",
          "age": 32,
          "gender": "male",
          "city": "bengaluru",
          "salary": 20000
   },
   {
          "firstname": "pradeep",
          "lastname": "reddy",
          "age": 21,
          "gender": "male",
          "city": "hyd",
          "salary": 25000
   },
   {
         "firstname": "mounika",
         "lastname": "mudiraj",
         "age": 20,
         "gender": "female",
         "city": "nalgonda",
         "salary": 30000
   },
   {
         "firstname": "nikhil",
         "lastname": "m",
         "age": 22,
         "gender": "male",
         "city": "guntur",
         "salary": 2000
   },
   {
         "firstname": "riya",
         "lastname": "bhadouria",
         "age": 14,
         "gender": "female",
         "city": "indore",
         "salary": 40000
   }
];
```

1. Print all the firstnames.
2. Print all the full names.
3. Print only those names whose age is more than 25.
4. Print all female names.
5. Print only those names whose salary is more than 25000 and increase their salaries by 15%.
6. Using prompt, print only those names whose city is "hyd".
7. Print the total salary of all the people.
8. Print all the **female** names.
9. Print all the **firstnames** whose salary is **more** than 25000.
10. Using prompt, print all names whose city is "**hyd**".
11. Print all the **fullnames** in the alphabetical order.
12. Print all the **fullnames** in the increasing order of their age.
13. Print all the **fullnames** in the reverse alphabetical order.
14. Print all the **fullnames** in the decreasing order of their salaries.
15. Print all the **cities** in which the people live. There should not be any **duplicate** cities.
16. Print all the **male** names whose age is greater than **25.**
17. Print all names that starts with "**p**" and the firstname should be in **UPPERCASE**. e.g. PRAVEEN Gubbala
18. Print the average age of all students

Advanced:

1. **Print all the full names in ascending order of their salaries.**

2. **Separate all male and female people and store them in separate arrays.**
3. **Store all the cities in an array. There should not be any duplicates.**
4. **Print all the full names with the first letter in uppercase. Example: "Praveen Gubbala".**
5. **Print the full names of all the male people in alphabetical order(a-z).**
6. **Print the full names of all the female people in reverse alphabetical order(z-a).**
7. **Increase the salaries of all people by 15% and print the new array of objects.**
8. **Check whether any person is having the letter 'k' in their first names.**
9. **Check whether all people are having the letter 'a' in their last names.**
10. **Add a "role" property to each object and print the new array of objects. For example, add "role":"Lead Engineer" to each object.**
11. Sort the array based on their lastname

Assignment2:

```
var products = [
        {
            "name": "Duracell - AAA Batteries (4-Pack)",
            "type": "HardGood",
            "price": 5.49,
            "category": "Household Batteries",
            "manufacturer": "Duracell",
        },
        {
            "name": "Hard Rock TrackPak - Mac",
            "type": "Software",
            "price": 29.99,
            "category": "Recording Equipment",
            "manufacturer": "Hal Leonard",
        },
        {
            "name": "Duracell - AA 1.5V CopperTop Batteries (4-Pack)",
            "type": "HardGood",
            "price": 5.62,
            "category": "Household Batteries",
            "manufacturer": "Duracell",
        },
        {
            "name": "Energizer - MAX Batteries AA (4-Pack)",
            "type": "HardGood",
            "price": 5.32,
            "category": "Household Batteries",
            "manufacturer": "Energizer",
        },
        {
            "name": "METRA - Antenna Cable Adapter - Black",
            "type": "HardGood",
            "price": 13.99,
            "category": "Antennas & Adapters",
            "manufacturer": "Metra",
        },
        {
            "name": "WipeDrive Six - Mac|Windows",
            "type": "Software",
            "price": 23.99,
            "category": "Maintenance Software",
            "manufacturer": "White Canyon",
        }
    ];
```

1. Print all the product names.
2. Print all the hardgoods.
3. Print all the softwares.
4. Print all the categories.
5. Print only the products manufactured by Duracell.
6. Print the product names in ascending order of their prices.
7. Print only those products whose price is more than 14.99.
8. Print only those products whose price is less than 9.99.
9. Print the total price of all the hardgoods.
10. Print the average price of the softwares.

Advanced level:

1. **Print all the product names in ascending order of their prices.**
2. **Separate all Hardgood and Software and store them in separate arrays.**
3. **Store all the categories in an array. There should not be any duplicates.**
4. **Print all the product names with numbering.**

   Expected output: "1. Duracell - AAA Batteries (4-Pack)"

   "2. Hard Rock TrackPak - Mac" ……..and so on.
5. **Print the product names of all the Hardgoods in alphabetical order(a-z).**

6. **Print the product names of all the Softwares in reverse alphabetical order(z-a).**
7. **Decrease the price of all products by 10% and print the new array of objects.**
8. **Check whether any of the manufacturers is "Metra".**
9. **Check whether all products are manufactured by "Duracell".**
10. **Add a "discount" property to each object and print the new array of objects. For example, add "discount": 1.5 to each object.**

Assignment 3:

```
var products = [
                    {
                        "id": 1,
                        "title": "iPhone 9",
                        "description": "An apple mobile which is nothing like apple",
                        "price": 549,
                        "discountPercentage": 12.96,
                        "rating": 4.69,
                        "stock": 94,
                        "brand": "Apple",
                        "category": "smartphones",
                        "thumbnail": "products/1/thumbnail.jpg"

                    },
                    {
                        "id": 2,
                        "title": "iPhone X",
                        "description": "6.5-inch Super Retina HD display with OLED",
                        "price": 899,
                        "discountPercentage": 17.94,
                        "rating": 4.44,
                        "stock": 34,
                        "brand": "Apple",
                        "category": "smartphones",
                        "thumbnail": "products/2/thumbnail.jpg"


                    },
                    {
```

```
        "id": 3,
        "title": "Samsung Universe 9",
        "description": "Samsung's new phone which goes beyond Galaxy",
        "price": 1249,
        "discountPercentage": 15.46,
        "rating": 4.09,
        "stock": 36,
        "brand": "Samsung",
        "category": "smartphones",
        "thumbnail": "products/3/thumbnail.jpg"

    },
    {
        "id": 4,
        "title": "OPPOF19",
        "description": "OPPO F19 is officially announced on April 2021.",
        "price": 280,
        "discountPercentage": 17.91,
        "rating": 4.3,
        "stock": 123,
        "brand": "OPPO",
        "category": "smartphones",
        "thumbnail": "products/4/thumbnail.jpg"

    },
    {
        "id": 5,
        "title": "Huawei P30",
        "description": "Huawei's re-badged P30 Pro New Edition",
        "price": 499,
        "discountPercentage": 10.58,
        "rating": 4.09,
        "stock": 32,
        "brand": "Huawei",
        "category": "smartphones"
```

```
            }
        ];
```

1. **Print all the product titles in ascending order of their prices.**
2. **Separate all the Apple and non-Apple products and store them in separate arrays.**
3. **Store all the brands in an array. There should not be any duplicates.**
4. **Modify all the descriptions so that they have only 20 characters and print the new descriptions.**
   Expected output: "An apple mobile whic"
   "6.5-inch Super Retin" ……and so on
5. **Print the product titles having price more than 500 in alphabetical order(a-z).**
6. **Print the product names having stock less than 50 in reverse alphabetical order(z-a).**
7. **Decrease the "discountPercentage" of all products by 3% and print the new array of objects.**
8. **Check whether any of the products has stock less than 10.**
9. **Check whether all products have thumbnails or not.**
10. **Add a "newPrice" property to each object by calculating the price according to the discountPercentage and print the new array of objects.**
    For example:
    "price": 549,
    "discountPercentage": 12.96,
    "newPrice": 71.1504……………………………………..(549-12.96% = 71.1504)

Assignment 4:

Here's a simple array of objects, which could represent a list of students with their names and ages:

```
const students = [
    { name: "John", age: 18 },
    { name: "Emily", age: 20 },
    { name: "Michael", age: 22 },
    { name: "Sarah", age: 19 },
 ];
```

Now, here are some assignment questions based on this array of objects:

Assignment Questions:

1. **Access Object Properties**: Write a function that takes the name of a student as input and returns the corresponding age of the student. If the student doesn't exist, return null.
2. **Array Length**: Write a function that returns the total number of students in the array.
3. **Find Oldest Student**: Write a function that finds the oldest student from the array and returns their name and age.
4. **Add a New Student**: Write a function that adds a new student to the array. The function should take the student's name and age as parameters.
5. **Remove a Student by Name**: Write a function that removes a student from the array based on their name. If the student doesn't exist, return a message saying "Student not found."
6. **Filter Students by Age**: Write a function that filters students who are older than a specified age. For example, filter out all students over the age of 20.
7. **Update Student's Age**: Write a function that takes a student's name and a new age as parameters and updates that student's age in the array. If the student doesn't exist, return a message saying "Student not found."
8. **Sort Students by Age**: Write a function that sorts the array of students by age in ascending order.
9. **Average Age**: Write a function that calculates and returns the average age of all students in the array.
10. **Check if Student Exists**: Write a function that checks if a student with a specific name exists in the array. Return true if the student exists, otherwise return false.

Assignment 5:

Here's a more complex array of objects, which represents a list of employees with additional attributes such as department, salary, and status:

```
    const employees = [
                    { id: 1, name: "John Doe", department: "Engineering", salary: 55000,
status: "active" },
                    { id: 2, name: "Jane Smith", department: "Marketing", salary: 60000,
status: "inactive" },
                    { id: 3, name: "Michael Johnson", department: "Engineering", salary:
75000, status: "active" },
                    { id: 4, name: "Emily Davis", department: "Human Resources", salary:
48000, status: "active" },
                    { id: 5, name: "Chris Lee", department: "Marketing", salary: 53000, status:
"ina ctive" },
                    { id: 6, name: "Sarah Brown", department: "Engineering", salary: 62000,
status: "active" },
                    { id: 7, name: "David Wilson", department: "Human Resources", salary:
47000, status: "active" },
                { id: 8, name: "Anna White", department: "Marketing", salary: 59000,
status: "inactive" }
   ];
```

Now, here are the assignment questions, sorted by difficulty level:

Beginner Level (Basic Array and Object Manipulation)

1. **Access Employee Data**: Write a function that takes an employee's ID as input and returns the employee's name and department. If the employee doesn't exist, return "Employee not found".
2. **Total Number of Employees**: Write a function that returns the total number of employees in the array.
3. **List Active Employees**: Write a function that returns an array of names of all employees who are currently "active."
4. **Find Employee by Department**: Write a function that takes a department name as input and returns a list of employees working in that department.

Intermediate Level (Advanced Array Methods and Filtering)

5. **Average Salary**: Write a function that calculates and returns the average salary of all employees in the company.
6. **Update Employee Status**: Write a function that takes an employee ID and a new status ("active" or "inactive") as parameters and updates the employee's status. If the employee doesn't exist, return "Employee not found".
7. **Remove Employee by ID**: Write a function that removes an employee from the array based on their ID. If the employee doesn’t exist, return "Employee not found".
8. **Employees with Salary Above a Threshold**: Write a function that takes a salary threshold as input and returns an array of employees who earn more than the specified salary.
9. **Sort Employees by Salary**: Write a function that sorts employees in descending order by their salary.

Advanced Level (Complex Array Operations)

10. **Department-wise Salary Calculation**: Write a function that calculates the total salary expense for each department. The result should be an object where the keys are department names, and the values are the total salaries for each department.
11. **Employee with the Highest Salary**: Write a function that returns the employee who has the highest salary, along with their name, department, and salary.
12. **Average Salary by Department**: Write a function that calculates and returns the average salary of employees in each department. The result should be an object where the keys are department names, and the values are the average salary for each department.
13. **Inactive Employees Count**: Write a function that returns the number of employees who have an "inactive" status.
14. **Update Salary Based on Department**: Write a function that increases the salary of all employees in a specific department by a certain percentage. The function should take the department name and percentage increase as parameters.
15. **Promote Employees (Based on Salary)**: Write a function that promotes employees who have a salary above a certain threshold (e.g., $60,000) by giving them a $10,000 raise. Return the updated list of employees with the new salary.

16. **Employee Report**: Write a function that returns a report of all employees in the following format: Name: [name], Department: [department], Salary: [salary], Status: [status]. The report should be sorted alphabetically by employee name.