Student Management System

Description:

In this CourseWrok, we will create a **Student Management System** for MedTech. After this you will be able to build a fully functional program.

Exercises:

1. Create a javascript file called ***StudentManagementSystem.js*** and open it in ***Your Chosen Code Editor.***

2. Declare a variable called ***numberOfStudents*** and give it the value zero.

3. Write a function called ***addStudent*** that takes no parameters and increases the ***numberOfStudents*** variable by one every time it is invoked.

```
function addStudent() {
  //Write your code here
}

console.log(numberOfStudents) -> 0
addStudent()
console.log(numberOfStudents) -> 1
addStudent()
console.log(numberOfStudents) -> 2
```

4. Write a function called ***getNumberOfStudents*** that takes no parameters and returns the variable ***numberOfStudents.***

```
function getNumberOfStudents() {
  //Write your code here
}

getNumberOfStudents() -> 0
addStudent()
getNumberOfStudents() -> 1
```

5. Declare an empty array called ***students.***

6. Modify the function ***addStudent*** so that it takes a string (name) as a parameter and adds that string to the array ***students.***

```
function addStudent(name) {
  //Write your code here
}

console.log(students) -> []
addStudent("Josh")
console.log(students) -> ["Josh"]
addStudent("John")
console.log(students) -> ["Josh", "John"]
```

7. Invoke the function ***addStudent*** ***5*** times with the names of 5 of your colleagues.

8. Write a function called ***clearStudents*** that takes no parameters and removes all the elements of the array and resets ***numberOfStudents*** to ***zero.***

```
function clearStudents(name) {
  //Write your code here
}

console.log(students) -> ["Josh", "John"]
getNumberOfStudent() -> 2
clearStudents()
console.log(students) -> []
getNumberOfStudent() -> 0
```

9. Modify the function ***addStudent*** so it assigns the length of the students array to the variable ***numberOfStudents.***

```
function addStudent(name) {
  //Write your code here
}

getNumberOfStudent() -> 0
console.log(students) -> []
addStudent("Josh")
console.log(students) -> ["Josh"]
getNumberOfStudent() -> 1
```

10. Write a function called ***createFullName*** that takes two parameters (firstName, lastName) and concatenates the firstName and lastName with an empty space in between and then returns the full name.

```
function createFullName(firstName, lastName) {
  //Write your code here
}

createFullName("Josh", "Bull") -> "Josh Bull"
createFullName("John", "Doe") -> "John Doe"
```

11. Invoke the ***addStudent*** 5 times using the function **fullName** as a parameter.

12. Write a function called ***getStudentByInitials*** that takes a string (name) and a character as a parameter and returns true if the string (studentName) starts with the character and false otherwise.

```
function getStudentByInitials(name, initial) {
  //Write your code here
}

getStudentByInitials("Josh", "J") -> true
getStudentByInitials("Josh", "A") -> false
```

13. Write a function called ***isFullName*** that takes a string (name) and returns true if the parameter consists of both a firstName and LastName.
HINT: Use .split()

```
function isFullName(name) {
  //Write your code here
}

isFullName("John Doe") -> true
isFullName("John") -> false
```

14. Delete the function ***getStudentByInitials.***

15. Write a function called ***getStudentsByInitials*** (notice the **S** in studentS) that takes a character as a parameter and loops through the array **students** and then returns an array of all the students whose firstName starts with that character.
NOTE: Use While Loop

```
function getStudentsByInitials(initial) {
  //Write your code here
}

console.log(students) -> [“John”, “Josh”, “Bill”]
getStudentsByInitials(“J”) -> [“John”, “Josh”]
```

16. Modify the function ***getStudentsByInitials*** so that it takes two characters as parameters and returns an array with students that have a fullName with their first name starting with the first initial and their last name starting with the second initial.
HINT: Use the function isFullName

```
function getStudentsByInitials(firstInitial,
secondInitial) {
  //Write your code here
}

console.log(students) -> [“John Doe”, “Joshua”, “Bill
Whethers”]
getStudentsByInitials(“J”, “D”) -> [“Nancy Ajram”]
```

17. Declare an object called ***student*** and give it the properties firstName, lastName, fullName, email, age & education.

18. Fill the object with your information.

19. Create a function called ***createStudent*** that takes 5 parameters (firstName, lastName, email, age & education) and returns a student object with those properties.

```
function createStudent(firstName, lastName, email, age,
education) {
  //Write your code here
}

createStudent("John", "Doe", "john.doe@gmail.com", 25,
"CS") -> [{ firstName: "John", lastName: "Doe", email:
"john.doe@gmail.com", age: 25, education: "CS" }]
```

20. Modify the function ***createStudent*** so that it adds the ***fullName*** property using the ***createFullName*** function.

```
function createStudent(firstName, lastName, email, age,
education) {
  //Write your code here
}

createStudent("John", "Doe", "john.doe@gmail.com", 25,
"CS") -> [{ firstName: "Nancy", lastName: "Shalaby",
fullName: "John Doe", email: "john.doe@gmail.com", age:
25, education: "CS" }]
```

21. Delete the ***student*** object you created.

22. Invoke the ***createStudent*** function with your information and save it in a variable called **me.**

23. Create a function called ***addSkills*** that takes a student object and an array of skills and adds a property called skills with the array as a value to the object.

```
function addSkills(student, skills) {
  //Write your code here
}

var student1 = createStudent("John","Doe",
"john.doe@gmail.com", 25, "CS")
console.log(student1.skills) -> undefined
```

```
addSkills(student1, [“HTML”, “CSS”])
console.log(student1.skills) -> [“HTML”, “CSS”]
```

24. Modify the function ***addSkills*** so that it only adds the new skills from the array; the skills property will not repeat values when adding skills to the object.

```
function addSkills(student, skills) {
  //Write your code here
}

var student1 = createStudent(“John”, ”Doe”,
“john.doe@gmail.com”, 25, “Music”)
addSkills(student1, [“HTML”, “CSS”])
console.log(student1.skills) -> [“HTML”, “CSS”]
addSkills(student1, [“Javascript”, “CSS”])
console.log(student1.skills) -> [“HTML”, “CSS”,
“Javascript”]
```

25. Clear the students array by invoking the function ***clearStudents.***

26. Delete the variable object called **me.**

27. Modify the ***addStudent*** function so it takes a **student** object as a parameter instead of a string and adds the object to the ***students*** array.

```
function addStudent(student) {
  //Write your code here
}

console.log(students) -> []
var student1 = createStudent(“John”, ”Doe”,
“john.doe@gmail.com”, 25, “CS”)
addStudent(student1)
console.log(students) -> [{ firstName: “John”, lastName:
“Doe”, fullName: “John Doe”, email: “john.doe@gmail.com”,
age: 25,education: “CS” }]
```

28. Invoke the function ***addStudent*** **5** times with your colleague’s information using the **createStudent** function.

29. Invoke the function ***addSkill*** for **3** of the students.

30. Write a function called ***removeStudent*** that takes a firstName as a parameter and loops through the student array and then removes the student with that name from the array.
NOTE: Use For Loop.

```
function removeStudent(firstName) {
  //Write your code here
}

console.log(students) -> [{ firstName: “John”, lastName: “Doe”, fullName: “John Doe”, email: “john.doe@gmail.com”, age: 25,education: “CS” }]
removeStudent(“Nancy”)
console.log(students) -> []
```

31. Modify the function ***removeStudent*** so that it assigns the array length to the variable **numberOfStudent** after removing the student.

```
function removeStudent(firstName) {
  //Write your code here
}

getNumberOfStudents() -> 1
removeStudent(“John”)
getNumberOfStudents() -> 0
```

32. Write a function ***isStudentOlderThan*** that takes two parameters, a student object and an age, and returns true if the student is older than that age.

```
function isStudentOlderThan(student, age) {
  //Write your code here
}

var student1 = createStudent(“John”, ”Doe”, “john.doe@gmail.com”, 25, “CS”)

isStudentOlderThan(student1, 18)-> true
isStudentOlderThan(student1, 27)-> false
```

33. Write a function called ***doesStudentHaveSkills*** that takes a student object and returns true if the student has at least one skill.

```
function doesStudentHaveSkills(student) {
  //Write your code here
}

var student1 = createStudent("John", "Doe",
"john.doe@gmail.com", 25, "CS")

doesStudentHaveSkills(student1) -> false
addSkills(student1, ["HTML"])
doesStudentHaveSkills(student1) -> true
```

34. Write a function called ***isStudentQualified*** that takes a student object as a parameter and returns true if the student is over 18 and has skills.
HINT: Use isStudentOlderThan and doesStudentHaveSkills.

```
function isStudentQualified(student) {
  //Write your code here
}

var student1 = createStudent("John", "Doe",
"john.doe@gmail.com", 25, "CS")
isStudentQualified(student1) -> false
addSkills(student1, ["HTML"])
isStudentQualified(student1) -> true
```

35. Write a function called ***numberOfStudentsOlderThan*** that takes age as a parameter and returns a number of all students that are older than that age.
HINT: Use isStudentOlderThan

```
function numberOfStudentsOlderThan(age) {
  //Write your code here
}

var student1 = createStudent("Ben", "Vega",
"ben.vega@gmail.com", 23, "RE")
addStudent(student1)

var student2 = createStudent("Josh", "Hep",
"josh.hep@gmail.com", 26, "CS")
addStudent(student2)

var student3 = createStudent("Jane", "Nash",
"jane.nash@gmail.com", 28, "CSE")
addStudent(student3)

numberOfStudentsOlderThan(25)-> 2
```

36. Write a function called ***getStudentsWithSkills*** that takes no parameters and returns an array of objects with all the students with the skills property.
HINT: Use doesStudentHaveSkills.

```
function getStudentsWithSkills() {
  //Write your code here
}

var student1 = createStudent("Nancy","Shalaby",
"x@gmail.com", 25, "Music")
addSkills(student1, ["HTML", "CSS"])
addStudent(student1)

var student2 = createStudent("Amjad","Hamawi",
"y@gmail.com", 19, "Art")
addStudent(student2)

getStudentsWithSkills()-> [{ firstName: "Nancy", lastName:
```

```
“Shalaby”, fullName: “Nancy Shalaby”, email:
“x@gmail.com”, age: 25, education: “Music”, skill:[“HTML”,
“CSS”] }]
```

37. Write a function ***getAllStudentsWithSkill*** that takes a string (skill) as a parameter then loops through all the students and returns an array of objects with all students who have that skill.

```
function getAllStudentsWithSkill(skill) {
  //Write your code here
}

var student1 = createStudent(“Nancy”, ”Shalaby”,
“x@gmail.com”, 25, “Music”)
addSkills(student1, [“HTML”, “CSS”])
addStudent(student1)

var student2 = createStudent(“Amjad”, ”Hamawi”,
“y@gmail.com”, 19, “Art”)
addSkills(student2, [“Javascript, “HTML”])
addStudent(student2)

getAllStudentsWithSkill(“CSS”) -> [{ firstName: “Nancy”,
lastName: “Shalaby”, fullName: “Nancy Shalaby”, email:
“x@gmail.com”, age: 25, education: “Music”, skills:
[“HTML”, “CSS”] }]

getAllStudentsWithSkill(“HTML”) -> [{ firstName: “Nancy”,
lastName: “Shalaby”, fullName: “Nancy Shalaby”, email:
“x@gmail.com”, age: 25,education: “Music”, skills:
[“HTML”, “CSS”] }, { firstName: “Amjad”, lastName:
“Hamawi”, fullName: “Amjad Hamawi”, email: “y@gmail.com”,
age: 19,education: “Art”, skills: [“Javascript”,
“HTML”] }]
```