**School Of computing**

**Practical 8:**

**Arrays of Objects**

**ST0502 Fundamentals of Programming**

**What you will learn / do in this lab**

* Write JavaScript functions that can take in or return an array
* Create an array of objects
* Write programs that utilize array of objects

You may open up your Visual Studio Code (VSC) and work from there for the remaining questions.

1. a) Create a **Square** class as shown below:

|  |
| --- |
| class Square{  constructor(length){  this.length = length;  }  calculateArea(){  return this.length \*\* 2;  }  } //end Square class  //main prgram starts here |

b) In the main program, declare an array named ***sqArray*** to store 10 instances of the **Square** class.

Within a **for** loop, create 10 instances of the Square class using 10 random integer values (10 - 20) as their length and store the 10 Square instances in ***sqArray***. Print out the length and area of all the elements in the array.

|  |
| --- |
| Hint : Use this to generate the random integer values  Math.floor(Math.random() \* 11) + 10 |

Your main program should produce the following output:

|  |
| --- |
| Length Area  15 225.0  11 121.0  19 361.0  19 361.0  12 144.0  11 121.0  19 361.0  17 289.0  13 169.0  10 100.0 |

class Square{

    constructor(length){

         this*.*length = length;

    }

*calculateArea*(){

*return* this*.*length \*\* 2;

    }

}

varSqArray=[]

*for*(vari=0;i<10;i++){

    randomNum = *Math.floor*(*Math.random*() \* 11) + 10

    SqArray[i]=new *Square*(randomNum)

}

*process.stdout.write*('Length\tArea')

*for*(vari=0;i<10;i++){

*console.log*(SqArray[i]*.*length + '\t' +SqArray[i]*.calculateArea*())

}

2. Create new js file and code the ***Square*** class (you may copy from the previous program and paste it here). In the main program,

a) Declare a function ***createSquareArray()*** that creates and returns an array of 10 Square objects.

|  |
| --- |
| function createSquareArray() {  var len;  var sqArray = [];  for(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_) {  len = 10 + Math.floor(Math.random() \* 11);  sqArray.push(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_);  }  return \_\_\_\_\_\_\_\_\_;  } //createSquareArray |

1. Declare a method ***printSquareArray()*** that takes in an array of Square as parameter. It produces the same output as Q1 above.

|  |
| --- |
| function printSquareArray(\_\_\_\_\_\_\_\_\_\_) {  :  :  :  } //printSquareArray |

c) Use the following code fragments to test your functions.

|  |
| --- |
| var myArray = createSquareArray();  printSquareArray(myArray); |

**Sample Output :**

|  |
| --- |
|  |

3. An ***AddressBook*** program consists of 2 classes, ***Contact*** and ***AddressBook*** classes.

1. Create ***Contact*** class which consists of 2 properties, ***name*** and ***mobileNumber*** and 1 method ***getContactDetails()*** which returns the contact details in the following format:

|  |
| --- |
| Name: *<<Name>>*  Mobile Number: *<<Mobile Number>>* |

1. Create AddressBook class. The class consists of the following:
2. 1 Property, ***myContact***. ***myContact*** is an array of ***Contact*** objects. The code fragment below illustrates how you can define this property in the constructor of the class:

|  |
| --- |
| constructor() {  this.myContact = []; //class property  this.myContact.push(new Contact("Homer", 98849959));  this.myContact.push(new Contact("Marge", 84774744));  this.myContact.push(new Contact("Lisa", 86994994));  this.myContact.push(new Contact("Maggie", 94775883));  this.myContact.push(new Contact("Bart", 88838848));  } //end constructor |

1. A method, called ***getNumberOfContacts()***, which simply returns the number of elements of ***myContact*** array.
2. A method, called ***getContactAt()*** which takes in an integer parameter called ***index*** and returns a contact object stored at the ***index*** position of ***myContact*** array. E.g. if the pass-in index is **2**, the contact object containing Lisa details will be returned.
3. A method, called ***searchContact()*** which takes in the name of the contact to be searched. The method returns the mobile number of the person. E.g. if the name passed in **Bart**, the method will return 88838848. If the contact is not found, the method simply returns the string “not found!”.
4. In the main program, instantiate an instance of ***AddressBook***, namely ***myAddressBook*** and show the following menu:

|  |
| --- |
| Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> |

1. ***showAllContacts()*** function: The function displays each contact from the ***myAddressBook*** object (1-by-1) as shown below:

|  |
| --- |
| -----------------------------  Contact 2 of 5  -----------------------------  Name: Marge  Mobile Number: 84774744  -----------------------------  (Press N for next contact, P for previous contact, X to exit)  >> |

Note that the options are not case-sensitive. When **N or n** is entered, the next contact is displayed and when **P or p** is entered, the previous contact is shown. The user enters **X or x** to go back to the main menu.

1. ***search()*** function: The function prompts the user to enter the name of the contact and invokes ***searchContact()*** method of ***myAddressBook*** object as shown:

|  |
| --- |
| Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> **2**  Enter the name of the contact: **homer**  homer's mobile number is 98849959  Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> |
|  |

Here’s the sample run of the program:

|  |
| --- |
| Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> **1**  -----------------------------  Contact 1 of 5  -----------------------------  Name: Homer  Mobile Number: 98849959  -----------------------------  (Press N for next contact, P for previous contact, X to exit)  >> **n**  -----------------------------  Contact 2 of 5  -----------------------------  Name: Marge  Mobile Number: 84774744  -----------------------------  (Press N for next contact, P for previous contact, X to exit)  >> **n**  -----------------------------  Contact 3 of 5  -----------------------------  Name: Lisa  Mobile Number: 86994994  -----------------------------  (Press N for next contact, P for previous contact, X to exit)  >> **p**  -----------------------------  Contact 2 of 5  -----------------------------  Name: Marge  Mobile Number: 84774744  -----------------------------  (Press N for next contact, P for previous contact, X to exit)  >> **x**  Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> **2**  Enter the name of the contact: **John**  John's mobile number is not found!  Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> **2**  Enter the name of the contact: **Bart**  Bart's mobile number is 88838848  Personal Addressbook  --------------------------  (1) Show All Contacts  (2) Search Contact  (3) Exit  >> **3**  Good Bye! |

**- END -**