1.]

```
> var add =(function (){
      var counter  = 0;
      return function () { return counter +=1;}
  })();
<- undefined
> add();
<- 1
> add();
<- 2
> add();
<- 3
>
```

Closures maintain state of all the variables.

2.]

a.) starting with “lion”

```
> function LionExists(input){
      var pattern= new RegExp('^lion');
      return pattern.test(input.toLowerCase());
  }
<- undefined
> LionExists("LIONS are sleepinh");
<- true
> LionExists("lion are sleepinh");
<- true
> LionExists("sheep and lion are sleepinh");
<- false
```

b.) ending with “cat”

```
> function CatExists(input){
      var pattern= new RegExp('cat$');
      return pattern.test(input.toLowerCase());
  }
<- undefined
```

```
> CatExists("sdfdafcat");
<- true
> CatExists("CATsfgfsg");
<- false
> CatExists("sgsfCAT");
<- true
```

c.) check a,b,c; where b occurring 1 or more times

```
> function ABC(input){
      var pattern= new RegExp('ab+c');
      return pattern.test(input.toLowerCase());
  }
< undefined
> ABC("adfafdadafabbbbbbbbbbbccccccccccccc");
< true
> ABC("aaaaaaa");
< false
> ABC("ac");
< false
> ABC("abc");
< true
> ABC("abbbbbbbbc");
< true
> ABC("cba");
< false
```

3.] check a,b,c exists and location of last index

```
> function FindLocationofABC(input){
      var patternA=new RegExp('a');
      var patternB=new RegExp('b');
      var patternC=new RegExp('c');
      input=input.toLowerCase();
      if(patternA.test(input))
          console.log('last a index at
  '+input.lastIndexOf('a'));
      else
          console.log("a ddoes not exists");
      if(patternB.test(input))
          console.log('last b index at
  '+input.lastIndexOf('b'));
      else
          console.log("b ddoes not exists");
      if(patternC.test(input))
          console.log('last c index at
  '+input.lastIndexOf('c'));
      else
          console.log("c does not exists");
  }


< undefined
> FindLocationofABC("Sumanto like badminton");
  last a index at 14                                    VM2814:7
  last b index at 13                                    VM2814:11
  c does not exists                                     VM2814:17
< undefined
> FindLocationofABC("Eat icecream and relax");
  last a index at 20                                    VM2814:7
  b ddoes not exists                                    VM2814:13
  last c index at 7                                     VM2814:15
< undefined
> FindLocationofABC("Sweets and chocolates are chill");
  last a index at 22                                    VM2814:7
  b ddoes not exists                                    VM2814:13
  last c index at 26                                    VM2814:15
< undefined
> FindLocationofABC("box are cuboid ");
  last a index at 4                                     VM2814:7
  last b index at 10                                    VM2814:11
  last c index at 8                                     VM2814:15
< undefined
```

4.]

a.) sort

```
> arr=[3,4,6,1,2,7,8,9,0];
< ▶ (9) [3, 4, 6, 1, 2, 7, 8, 9, 0]
> arr.sort();
< ▶ (9) [0, 1, 2, 3, 4, 6, 7, 8, 9]
```

b.) multiply each element by 10

```
> arr.forEach(function(item, index, theArray) {
    theArray[index] = item*10;
  });
< undefined
> arr
< ▶ (9) [0, 10, 20, 30, 40, 60, 70, 80, 90]
```

\

c.) return numbers that are divisible by 3

```
> arr.filter(x => x%3 == 0);
< ▶ (4) [0, 30, 60, 90]
```

5.]

== → Ignores data type of the variables while comparing

=== → Data type is also taken into consideration while comparing

```
> 3=="3";
< true
> 3==="3";
< false
> true=="1";
< true
> true==="1";
< false
> "string" == new String ("string");
< true
> "string" === new String ("string");
< false
```