UNIT NO 2

Array , Function and String

14 Marks

Array

- The **Array** object lets you store multiple values in a single variable.
- It stores a fixed-size sequential collection of elements of the same type.
- An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Declaring an Array

1. **By array literal**

Syntax:

var arrayname=[value1,value2,.....valueN];

example

Declaring an Array

1. **By creating instance of array**

Syntax:

```
var arrayname= new Array();
```

new keyword is used to create instance of array.

example

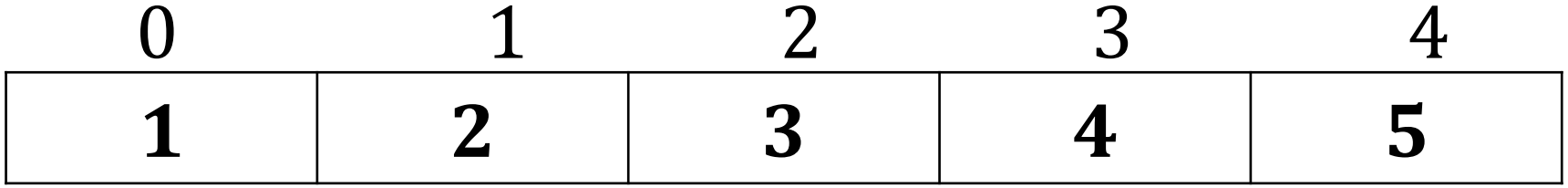
Initializing an Array

- Initialization is the process of assigning value of an array.
- All the elements should be placed in parenthesis and separated by comma.
- Example-

1. var a=[1,2,3,4,5];
2. var b= new Array(“c”, “c++”, “Java”, “VB”);

Defining an Array Elements

- Array contains a list of elements; each element in the array is identified by its index.

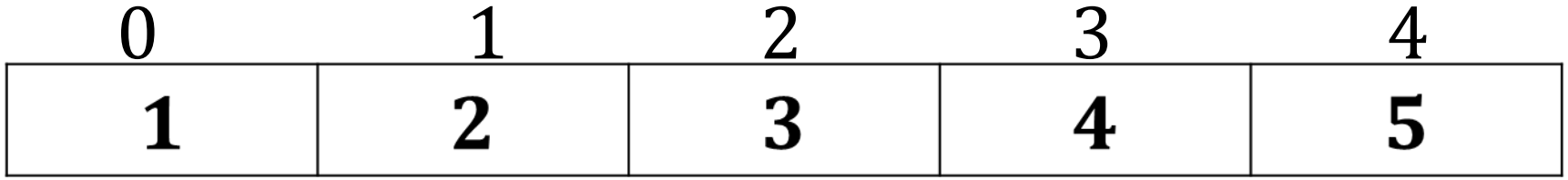


Array length=5
First index=0
Last index=4

- Assignment (=) operator is used to assign values to an array elements.
- Elements can be retrieving by its index positions.

Looping an Array

- **For Loop** iterate over each item in an array.
- Starting *or* First element is reference with an index of 0.
- Ending *or* Last element is array length minus 1.
- Elements in an array accessed by numeric index.



Array length=5
First index=0
Last index=4 (array length - 1)

Adding an Array Element

1. **Using push() method**
2. **Using unshift () method**

Using push() method-It adds one or more elements at the **end** of array.

Syntax-

array.push(element1,element2,……..element);

Example-

```
a.push("Java");
a.push(1,2);
a.push("Java", “PHP”, “CSS”);
```

Adding an Array Element

1. **Using push() method**
2. **Using unshift () method**

Using unshift() method-It adds one or more elements at the **beginning** of array.

Syntax-

array. unshift(element1,element2,……..element);

Example-

a. unshift("Java");
a. unshift(1,2);
a. unshift("Java", “PHP”, “CSS”);

Removing an Array Element

1. **Using shift() method**
2. **Using pop() method**

Using shift() method-It removes **first elements** of array.

Syntax-

array. shift();

Example-

a. shift();

Removing an Array Element

1. **Using shift() method**
2. **Using pop() method**

Using pop() method-It removes **last elements** of array.

Syntax-

array. pop();

Example-

a. pop();

Adding & Removing an Array Element

1. Using splice() method

Using splice() method-It add or remove elements from array.

Syntax-

array. splice(start,delete,element1,element2,……elementN);

Start-it represent index from where the **method start to extract** the element.
Delete-it is optional.it represent number of elements **to be removed**.
Element-it is optional.it represent the elements **to be inserted.**

Example-

a.splice(2,0, “Java”, “PhP”);

Sorting an Array Element

1. **Using sort() method**
2. **Using reverse() method**

sort() method-It sorts the elements of an array and returns the sorted array.

Syntax-

array. sort(compareFunction);

- Array will be sorted according to the compare() Function.

1. **compareFunction(function(a,b) {return a-b}) -----→Ascending**
2. **compareFunction(function(a,b) {return b-a}) -----→Descending**

- ✓ If the result is negative a is sorted before b.
- ✓ If the result is positive b is sorted before a.
- ✓ If the result is 0 no changes are done with the sort order of the two values.

7	2	5

Sorting an Array Element

1. **Using sort() method**
2. **Using reverse() method**

reverse() method-It sorts the elements of an array in reverse order i.e in descending order.

Syntax-

```
array. reverse();
```

Example-

```
var r=a.sort();
r.reverse();
```

Combining an Array Element into String

1. **Using join()**
2. **Using concat()**

join() –This function join all elements of an array into string.

Syntax-

array. join(separator);

Example-

var r=a.join();

- ✓ Separator-It is string like **/ , - _ *** etc.
- ✓ If separator is not mentioned ,it will take comma as separator.

Combining an Array Element into String

1. **Using join()**
2. **Using concat()**

concat() –This function join two or more array together.it returns new string which is combination of different string passed to it as arguments.

Syntax-

string1. concat(string2, string2,……. stringN);

Example-

var r=a.concat(“VB”);

Object as Associative Array

- Associative arrays are **dynamic objects** that the user **redefines as needed.**
- When you assign **values to keys** in a variable of type Array, the array is **transformed into an object**, and it loses the attributes and methods of Array.
- We can create it assigning a **literal to a variable**.
- **Syntax-**

array={key1:'value1', key2:'value2', key3:'value3'};

Function

- A function is a **subprogram** designed to perform a **particular task**.
- Functions are executed when they **are called**. This is known as **invoking a function.**
- **Values** can be passed **into functions** and **used** within the function.
- Functions always **return a value**. In JavaScript, if no return value is specified, the function will return **undefined**.
- Functions are **objects**.

Defining a Function

- The function definition consist of following parts:

1.The **function** keyword with **Name of function.**
2.A list of **parameters** names enclosed in parentheses.
3.The **statement enclosed** in braces.
4.The **return** key word (optional).

- Syntax :

```
function name(parameters)
{
Statements
}
```

Adding an Argument

- When one or more **variables** that are declared **within the parentheses** of a function definition is known as **arguments.**
- These arguments are used to **hold data** to perform **some task.**

function name(parameters)

function add (no1,no2)

Scope of Variable and arguments

- Global Scope:

  All the variables that you declare, is by default defined in global scope.

  When a variable is declared **Outside a function** with a var keyword then that variable is a **global variable** because it is available through **all parts of script**.

- Local Scope :

  All variable which are declared using var keyword **within function** then those variables are known as **local variables** and they are accessible **within function only.**

Calling a Function

Calling a function from HTML

- Functions can be called from HTML code in response of any particular event like page load, page unload, button click etc.

1. Write a function for a event.
2. Call a function on event.

Function Calling another Function

- We can call one function inside another function.
- The **Function** which calls **another Function** is called **Calling Function** and **function** which is called by **another Function** is call **Called Function.**

Returning a Value from Function

- The **return** keyword **stops the execution** of function and the value is return from function to the **function caller.**
- **Syntax**

```
return value;
```

String

- String are used for storing and manipulating text.
- String is zero or more characters written inside quotes.
- var c="india";
- var fname="Bob";
- var fname=new String("Bob");

Manipulate a String

- String manipulation means operations on string.
- The operations are :
  - ✓ **concatenation**
  - ✓ **extraction of sub string,**
  - ✓ **finding length,**
  - ✓ **finding position of character,**
  - ✓ **changing case of string character etc.**

Joining a String

- String concatenation means joining two string to create a new string by placing the copy of second string behind a copy of first string.
- **Methods:**

1. Using concatenation (+) operator:
   - **Syntax:** string1+ string2;
   - **Example:** var name=fname+sname;

1. **Using concat()**
   - **Syntax:** concat(str1,str2,…)
   - **Example:** var msg="India";
     var final=msg.concat("is my country");

Retrieving a Character from given Position

- charAt():returns the character at the ‘x’ position within the string.
- **Syntax:** String.charAt(x)
- **Example:**

```
var mystr=“Hello world”;
mystr.charAt(7);
```

Diving Text

- **Split():**This function is used to split the given string into array of strings by separating it into substrings using a specified separator.
- **Syntax-**string.spilit(separator, limit)
- The ***separator*** specifies the points where the split has to take place. If the separator is not specified then the entire string becomes one single array element. If the separator is an empty string (‘ ’) then every character of the string is separated by commas.
- The ***limit*** specifies the upper limit on the number of splits to be found in the given string.
- This function returns an array of strings that is formed after splitting the given string with the help of separator.

Retrieving a Position of Character in a String

1. indexOf()

- indexOf():the function searches and return the index number of the character or substring within the string .
- **Syntax:** indexOf(substr,[start])

substr is a string/character that we want to search and start is an optional argument specify the position within string to being the search. default value for **start** is 0.

- **Syntax:** string.indexOf(char/substring)

Retrieving a Position of Character in a String

2.lastindexOf()

- **lastindexOf():** the function searches and return the index number of last occurrence of the character or substring within the string . Searches the string from end to beginning.
- **Syntax:** string.lastindexOf(substr,[start]);
- The value of ***start*** can be between 0 to the length of string.

Retrieving a Position of Character in a String

3.search()

- **search():** the method searches the string for a specified value and return the position of the match found.

- **Syntax:** string.search(word);

- **Example:** var str="Hello world";
  str.search("world");

Copying a Sub-String

1.substring()

- **substring():** Returns the character in a string between “from” and “to” indexes.
- “to” is optional. and if it is omitted then it will search up to the end of the string.
- **Syntax:**string.substring(from,[to]);

Copying a Sub-String

2.substr()

- **substr():** Returns the character in a string beginning at "start" and through the specified number of character, "length".
- "length" is optional. and if it is omitted ,up to the end of the string is assumed.
- **Syntax:**string.substr(start,[length]);

Converting String to Number & Numbers to String

1.parseInt()

- **parseInt():** The parseInt parses a string and returns a whole number.

2.parseFloat()

- **parseFloat():** The parseFloat parses a string and returns a number.

3.Number()

- **Number():** The function converts the string to a number. if the string value is number then it will convert otherwise will return NaN as output.

4.toString()

- **toString ():** This function used to convert number to string.

Changing the case of String

1.toUpperCase()

- **toUpperCase():** The function will return the string with all of its characters converted to Uppercase.
- **Syntax:** string. toUpperCase();

2.toLowerCase()

- **toLowerCase():** The function will return the string with all of its characters converted to Lowercase.
- **Syntax:** string. toLowerCase();

Finding the Unicode of a Character

What is Unicode…???

the Unicode standard provides a unique number for every character, no matter what platform, device, application or language.

Example- 65 is an English letter “A”.

1.charCodeAt()

- **charCodeAt():** The function will return the Unicode value of the character at specified position.
- **Syntax:** string. charCodeAt(x);

2.fromCharCode()

fromCharCode():This method convert a given Unicode number into a character.

- **Syntax:** string.fromCharCode(n1,n2,……..nX);

m⊥>

ASCII	Hex	Symbol	ASCII	Hex	Symbol	ASCII	Hex	Symbol	ASCII	Hex	Symbol
64	40	@	80	50	P	96	60	`	112	70	p
65	41	A	81	51	Q	97	61	a	113	71	q
66	42	B	82	52	R	98	62	b	114	72	r
67	43	C	83	53	S	99	63	c	115	73	s
68	44	D	84	54	T	100	64	d	116	74	t
69	45	E	85	55	U	101	65	e	117	75	u
70	46	F	86	56	V	102	66	f	118	76	v
71	47	G	87	57	W	103	67	g	119	77	w
72	48	H	88	58	X	104	68	h	120	78	x
73	49	I	89	59	Y	105	69	i	121	79	y
74	4A	J	90	5A	Z	106	6A	j	122	7A	z
75	4B	K	91	5B	[	107	6B	k	123	7B	{
76	4C	L	92	5C	\	108	6C	l	124	7C	|
77	4D	M	93	5D	]	109	6D	m	125	7D	}
78	4E	N	94	5E	^	110	6E	n	126	7E	~
79	4F	O	95	5F	_	111	6F	o	127	7F	□