MVC: Model View Controller

1. **View** - Frontend
    1.1. Html5
    1.2. Css3
    1.3. Js
    1.4. Bootstrap
    1.5. Jquery
    1.6. Angular
    1.7. React js
2. **Controller** - Backend
    2.1. Php
    2.2. Java
        2.2.1. Core java
        2.2.2. Advanced java
        2.2.3. Spring Boot
    2.3. Python
    2.4. .net
    2.5. c#
    2.6. R
3. **Model** - Database Connectivity
    3.1. Mysql
    3.2. Postgre

SDLC - Software Development Life Cycle

1. Requirement gathering & Requirement Analysis - User Stories
    1.1. User Requirement Specification
    1.2. Customer Requirement Specification
2. Design - Architects
    2.1. High level Design
    2.2. Low level Design
3. Development - Java, Python, R - White box testing
4. Testing
5. Deployment
6. Maintenance

Java: Language

Computer - binary Language (1,0)
Int age = 23; - keyword identifier = value;

1. Compilers - File
2. Interpreter - Line to Line

C, C++ - High level Languages - Compilers
Java, Python - Compilers & Interpreters
Js - interpreters

HTML5: Hyper Text Markup Language

Hyper Text - Text Within a Text / Text about a text

Syntax: <h1>HELLO</h1>

- <h1> - Open Tag
- HELLO - Content
- </h1> - Close Tag
- H1 - Element

<!DOCTYPE HTML>

<html>

<head>
</head> - Meta data (Data about data)

<body>
</body>

</html>

Setup:

- Editor - Notepad, Notepad++, IDE (VSCode, Sublime text)
- Browser

VSCode download → System Installer

Index.html

Shift + 1 (Enter)

Extensions: Line Server → Ritwick Dey
Go Live

<h1>---</h1>
<h2>---</h2>
<h3>---</h3>
—----------
—----------
<h6>---</h6>
<h7>---</h7>

Lorem5 (Enter)

Right click → Open with Live server

<img src=”-----” alt=”” width=”200px” height=”200px”>

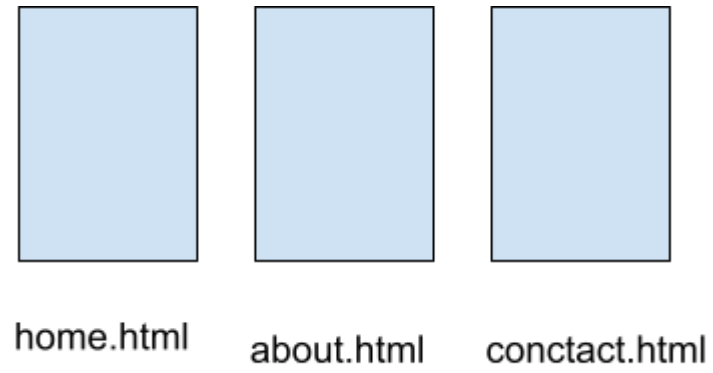
- Src - Property name

- —- - Property Value

Create Folder img

- Absolute path position (root) - From Google
- Shortcut path position (Relative) - img/1.jpg

Assignment-1:
AP Districts → 5
Each page for each district → Description → Paragraph



Website - Collection of pages
home
Href → Hyper refer
target="_blank" → New tab
download ="sample"
Mail to

List:

1. Order list -
2. Unordered list -
3. Description list - <dl></dl> <dt></dt> <dd></dd>

Order list:

```
<ol>
        <li>
        </li>
</ol>
```

```
<ol type="A">
<ol type="a">
<ol type="i">
<ol type="I">
```

Unorder list:

```
<ul>
</ul>
```

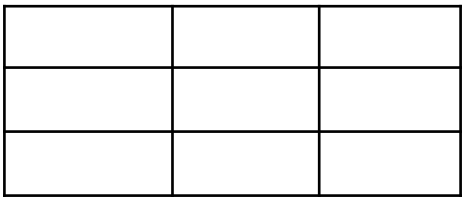
```
<ul type="circle">
```

<ul type=”square”>
<ul type=”none”>

Description list:

```
<dl>
        <dt>
                <dd>
                </dd>
        </dt>
</dl>
```

Table:

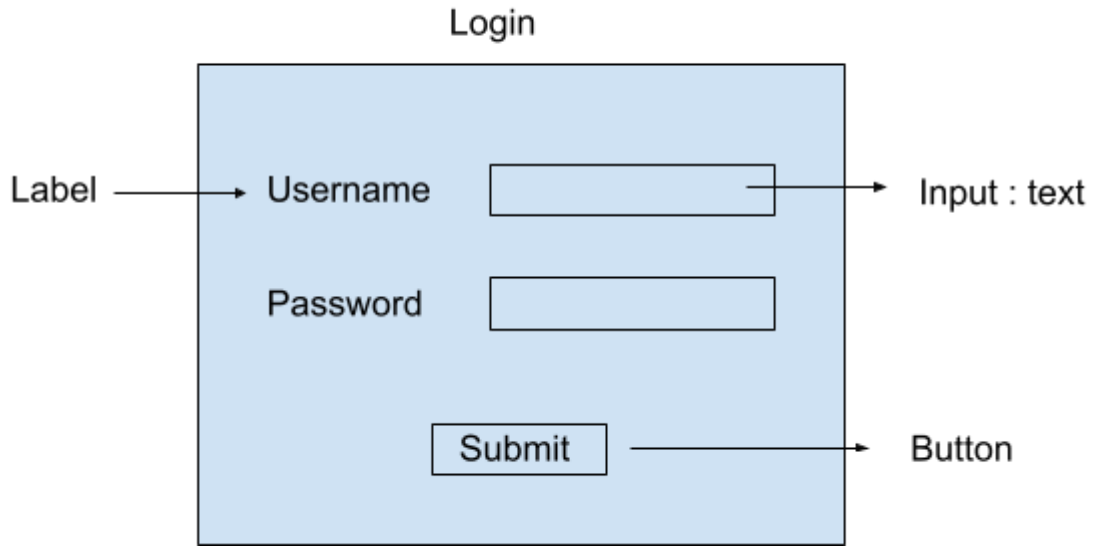


```
<table border = “2”>
        <tr>
                <th colspan = “2” rowspan = “2”>
                </th>
        </tr>
        <tr>
                <td>
                </td>
        </tr>
</table>
```

Forms:

Input:

- Text
- Radio
- Checkbox
- Dropdown
- Label



For = “ “ id = “ “
<form action = “”>
<input type = “text”>
<label for = “un”>User Name:</label>
<input type = “text” id = “un” placeholder = “Enter Username”>

<label for = “pwe”>Password:</label>
<input type = “Password” name = “ “ placeholder = “Enter Password”>

<input type = “Submit” value = “Submit”>
<label for = “gn”>Gender:</label>
<input type = “radio” name = “ “ id = “gn”>Male
“
“

<label for = “email”>Email</label>
<input type = “email” name = “ “ id = “email” placeholder = “Enter Email”>

<label for = “Country”>Country</label>
<input type = “Checkbox” name = “ “ id = “Country”>IND

<label for = “State”>State</label>
<select name = “ “ id = “state”>
	<option value = “select”>Select</option>
	<option value = “ap”>AP</option>
</select>
<label for = “dob”>DOB:</label>
<input type = “date” name = “ “ id = “dob”>

<label for = “Upload”>Upload</label>
<input type = “file” name = “ “ id = “upload”>

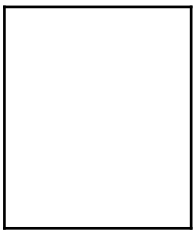
<input type = “search” name = “ “ id = “ “>

CSS:

Element_Selector
{
	Property name: property value;
	“
}

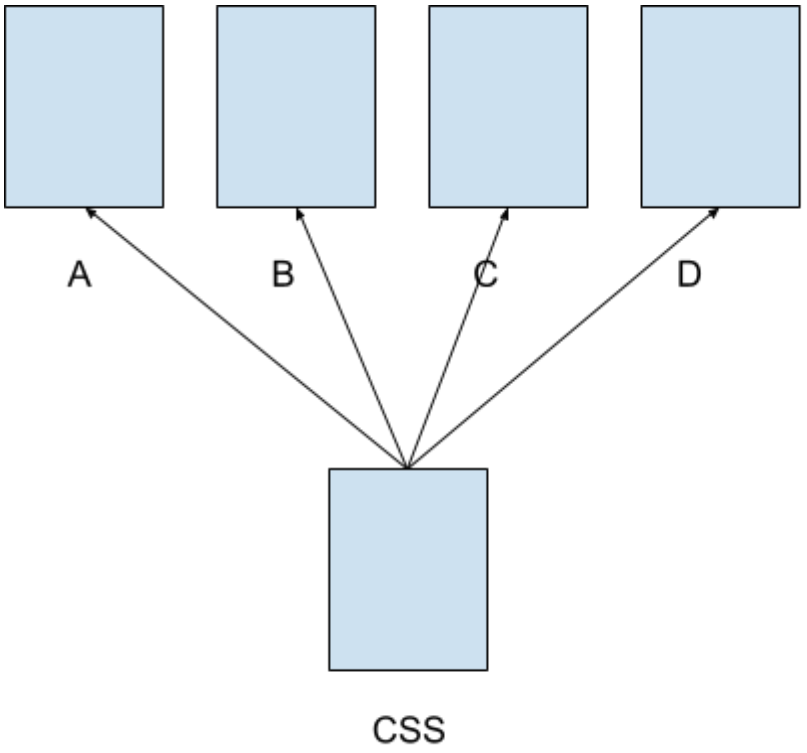
<h1>Hello</h1>
H1{
	Color:
	Fontsize:
}

- **Inline** - <h1 style = “color:red”>HTML</h1>
- **Internal**



home.html

- **External**



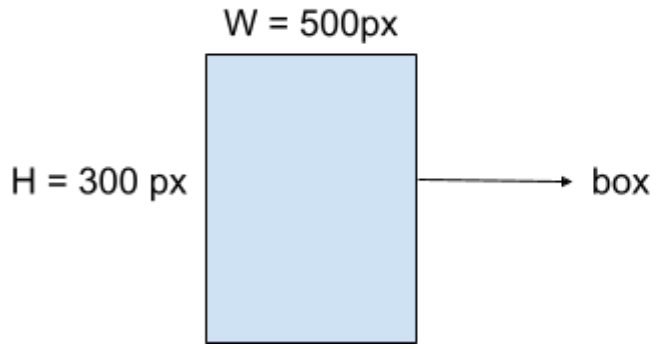
<h1 style = “color: red; text-align:center;”>....</h1>
<style>
	H1{

```
                Color:blue;
                Text-align:center;
        }
</style>
```

Sty.css:

```
Body{
        Background-color:burlywood;
}
<link rel = “stylesheet” href = “sty.css”>
```

Div:



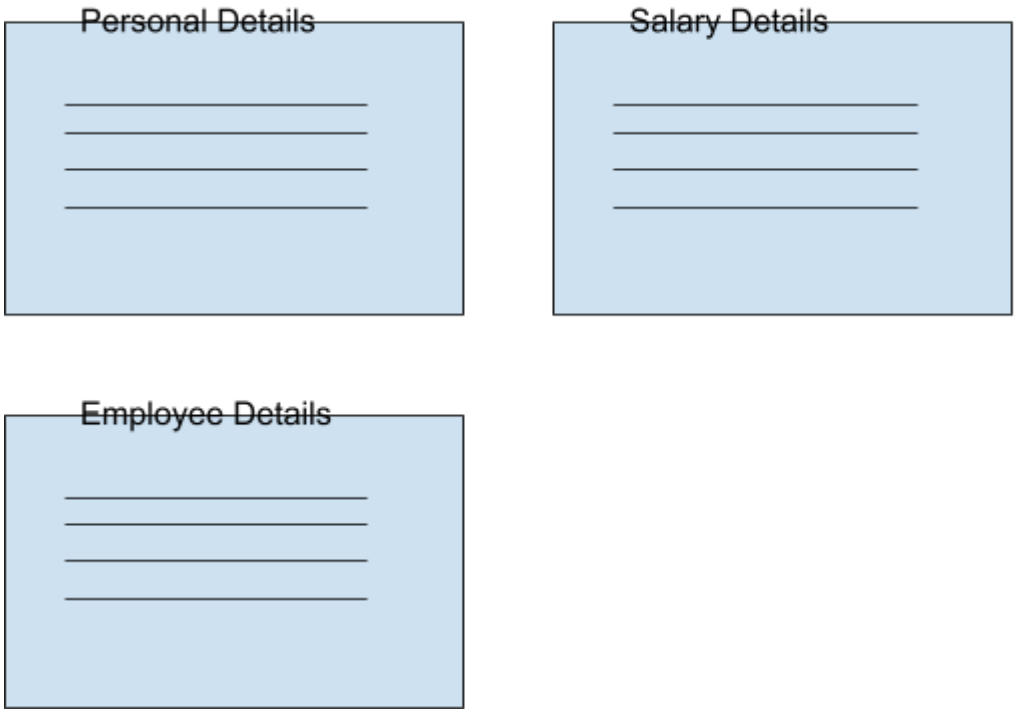

```
<div>
```

1. Element name (div)
2. Id (#)
3. Class (.)
4. *

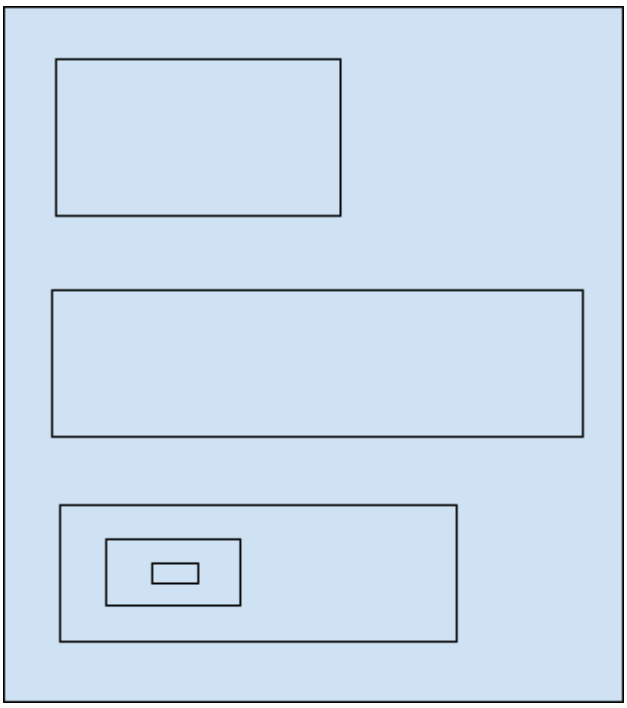
```
Div{
        Width: 300px;
        Height: 400px;
        Background-color: red;
}

#box1{
        Width: 200px;
        Height: 300px;
        Background-color: rebeccapurple;
}

#box2.box21{
        Width: 400px;
        Height: 200px;
        Background-color: crimson;
}

<div></div>
<div id = “box1”></div>
```

Assignment-2:
Table: Online Billing
Sample billing slips
Invoice - 2

Forms:



Container:



```
<form action = " ">
<fieldset>
        <legend>Personal Details</legend>
        <label for = "fn">First Name</label>
        <input type = "text" id = "fn"
```

Background:

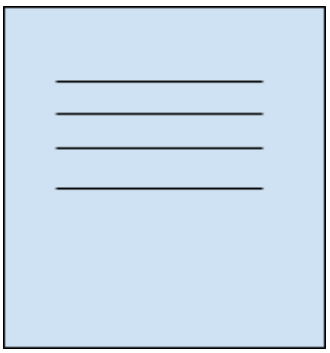
- Bg-color
    - Color-name
    - RGB(0-255)
    - Hexa(#FFFF)
- Bg-image
    - Position
    - Size
- Linear gradient
- Blend

```
<div class = "Container">

<style>
.container{
        Width: 50%;
        Height: 400 vh;
        Background-color: chocolate;
}

background-image:url(img/2.jpeg);
Background-position:center;
Background-size:cover;
```

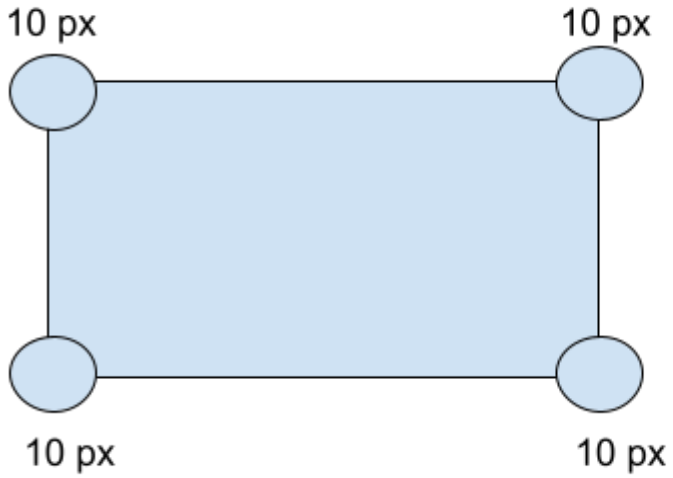
Background-repeat:no-repeat;
background:linear-gradient(to left/right, rgba(0,0,0,0.6), rgba(0,0,0,0.6), url(img/));
0.6 → 0.0 - 0.9
Background-repeat: repeat-x/y;
Background-position: center;
Background-size: cover;

Background-color: blueviolent;
Background-image: url(img/5.jpg);
Background-position: center;
Background-size: cover;
Background-blend-mode: lighten/darken/difference;

Border:



1. Border-style
2. Border-width
3. Border-color
4. radius




```
<style>
P{
        Width: 300px;
        Height: 300px;
        Background-color: aqua;
```

```
        Border-style: double;
        Border-width: 5px;
        Border-color: red;
        Border-radius: 10px/20px; → Pick opposite value
        Border-radius: 50%;
}

Img{
        Width: 400px;
        Height: 400px;
        Border-style: outset;
        Border-width: 10px;
        Border-color: chocolate;
        Border-radius: 30px;
        Border-radius: 50%;
}
}
```

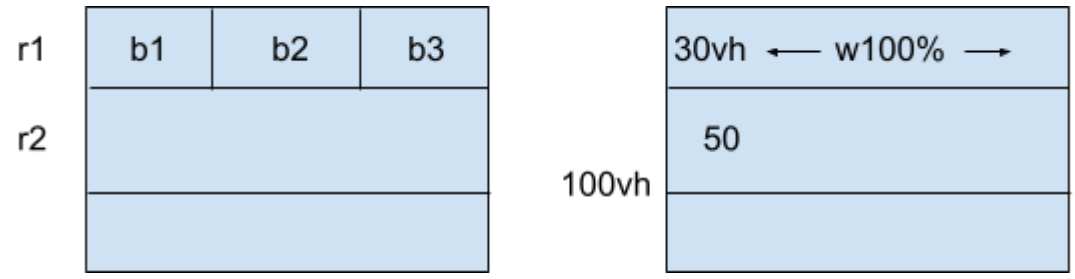
Display:

```
<h1>HELLO</h1>          HELLO
<h1>Hi</h1>             Hi
<h1>WELCOME</h1>        WELCOME

HELLO       Hi      WELCOME
```

1. Inline - Cannot define width & height
2. Inline-block - can define “
3. Block
4. None
5. Flex-box

```
<style>
P{
        Display: inline;
        Width: 150px;
        Background-color: aqua;
        Height: 20px;
}

Display: inline-block;

Display: block;

Display: none;

<div class = “container”>
        <div class = “block1”>
```

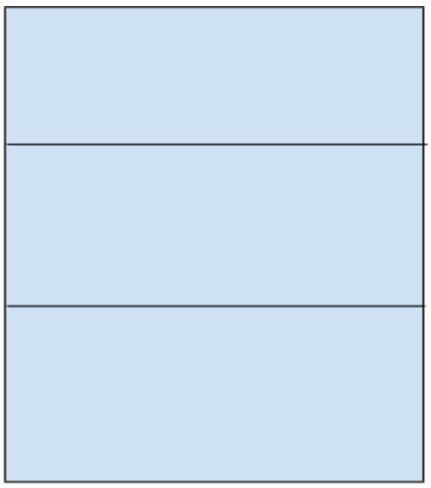
```
        </div>
</div>
```



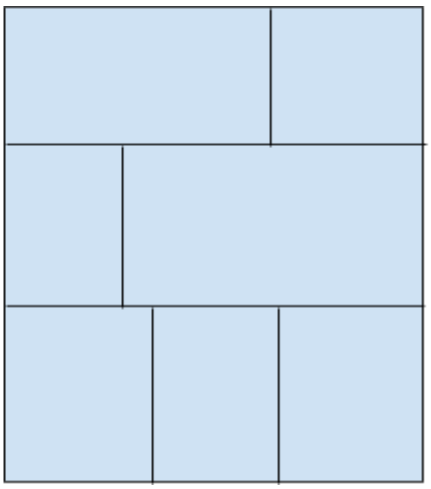

```
<div class = “container”>
<div class = “block1”>
.box*3
<div class = “box1”>box1</div>
<div class = “box2”>box2</div>
<div class = “box3”>box3</div>

<style>
.container{
        Width: 100%px;
        Background-color: aqua;
}
.block1{
        Width: 100%;
        Height: 300vh;
        Background-color: brown;
        Display: flex;
        Justify-content: space-between/space-evenly;
}
.block1.box1{
        Width: 30%;
        Height: 30vh;
        Background-color: blue;
}
.block1.box2{
        Width: 30%;
        Height
```

Assignment-3:

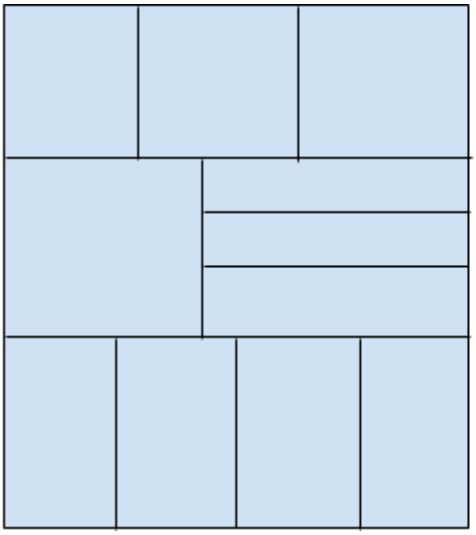
1)



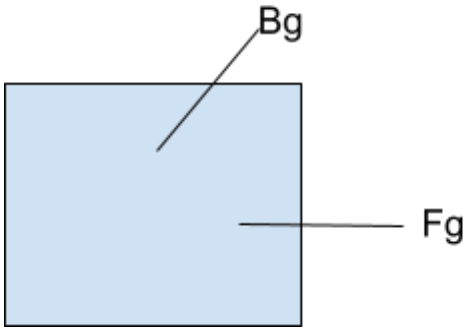
2)



3)



Background:



Bg-color:

- Color
    - Color-name
    - rgb(0-255)
    - hexacodes(#F8F8)
- bg-linear((),()),img()
- bg-img

```
<div class = “Container”>
<img src = “img/1.jpg” alt = “”>
</div>

<style>
.container{
        Width: 50%;
        Height: 50vh;
        Background-color: antiquewhite;
}
Img{
        Width: 300px;
        Height: 300px;
}

Background-color: #F8F8F8;
background:linear-gradient(to right/left rgba(0,0,0,0),rgba(0,0,0,0));
Alpha - (0-0.9(1))
background-image:url(img/2.jpg);
Background-repeat:no-repeat;
Background-position:center;
Background-size:cover;
Background-color:brown;
Background-blend-mode:lighten;
background:linear-gradient(rgba(0,0,0,0.6),rgba(0,0,0,0.6)),url(img(2.jpg);
```

Borders:

- Border-style
- Border-color
- Border-width
- border-radius

```
<div class = “container”>
<p>....</p>
<img src = “img/1.jpg” alt = “”>

<style>
.container{
        width:100%;
        Height:100vh;
}
P{
        Width:400px;
        Height:200px;
        Border-style:double;
        Border-color:brown;
        Border-radius:10px 20px;
```

```
}
Img{
        Width:300px;
        Height:300px;
        Border-style:groove;
        Border-width:10px;
        Border-color:greenyellow;
        Border-radius:circle;
}
```

Display:

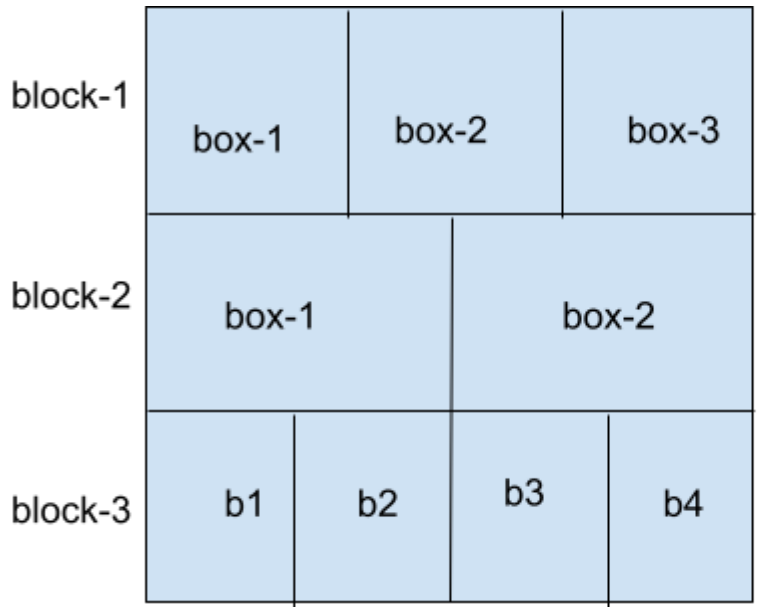
```
HTML  <p>HTML</p>
CSS     <p>CSS</p>
JS       <p>JS</p>
```

```
HTML  CSS     JS
```

1. Inline
2. Inline-block
3. Block
4. None
5. flex

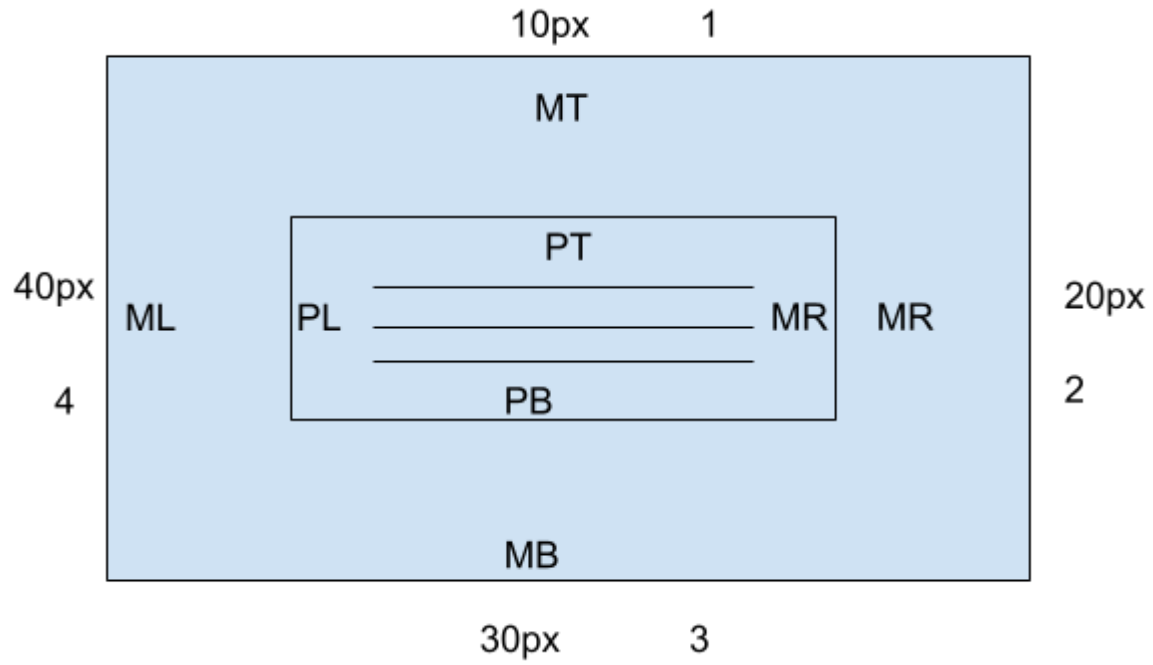
```
<div class = “container”>
<p>html</p>
<p>css</p>
<p>java</p>
<p>python</p>

<style>
.container{
        width:100%;
        Height:100vh;
}
P{
        Background-color:aqua;
        Display:inline-block;
        Width:150px;
        Height:30px;
}
```

Flex:




```
<div class = “container”>
<div class = “block-1”>block-1</div>
        <div class = “box-1”>box-1</div>
        <div class = “box-2”>box-2</div>
        <div class = “box-3”>box-3</div>
<div class = “block-2”>block-2</div>
<div class = “block-3”>block-3</div>

<style>
.container{
        width:100%;
        Height:100vh;
        Background-color:aqua;
}
.block-1{
        justify-content:space-around;/between/evenly
        width:100%;
        Height:
        Display:flex;
}
.block-1.box-1{
        width:30%;
        Height:30vh;
        Background-color:darkolivegroove;
}
```

Margin & padding:

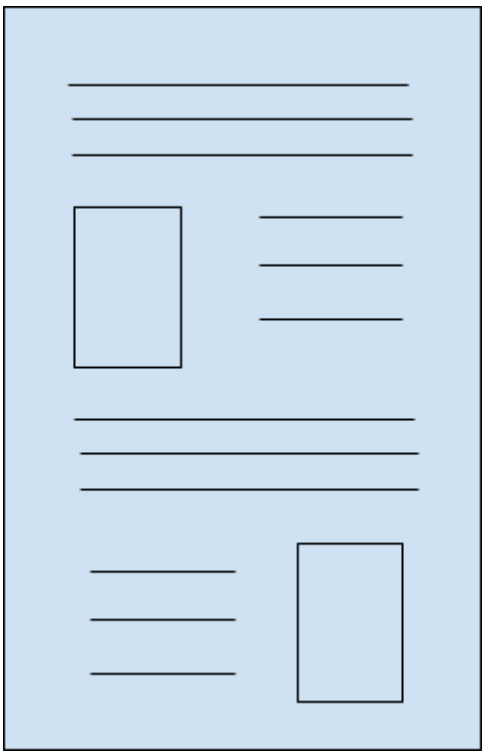


Margin: 10px 20px 30px 40px

...

<style>
.container{
}
P{
Border:solid 5px red;
Width:400px;
Height:200px;
Margin-left:10px;
Margin-top:
Margin:20px 50px 60px 100px;
Padding:20px;
}

Float:



...
<img src = “img/3.jpg” alt = “” width = “150px”height

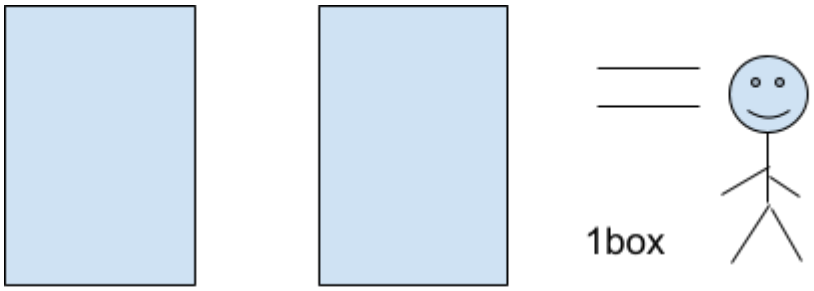
```
<style>
#img1{
        Width:
        Height:
        Float:left;
}
#img2{
        Margin:20px;
}
```

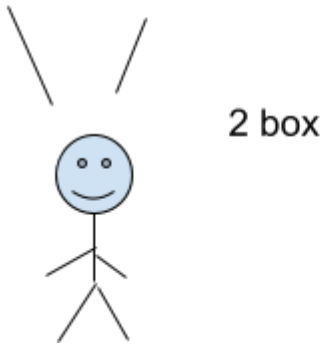
Font:
<h3>...</h3>
<h4>...</h4>

```
<style>
H3{
        Font-family: sans-serif;
        Font-size: 2.1rem;
        Font-style: italic;
        Color: brown;
}
H4{
        Font-family:serif;
        Font-weight:900;
        font-variant:small-caps;
```

}

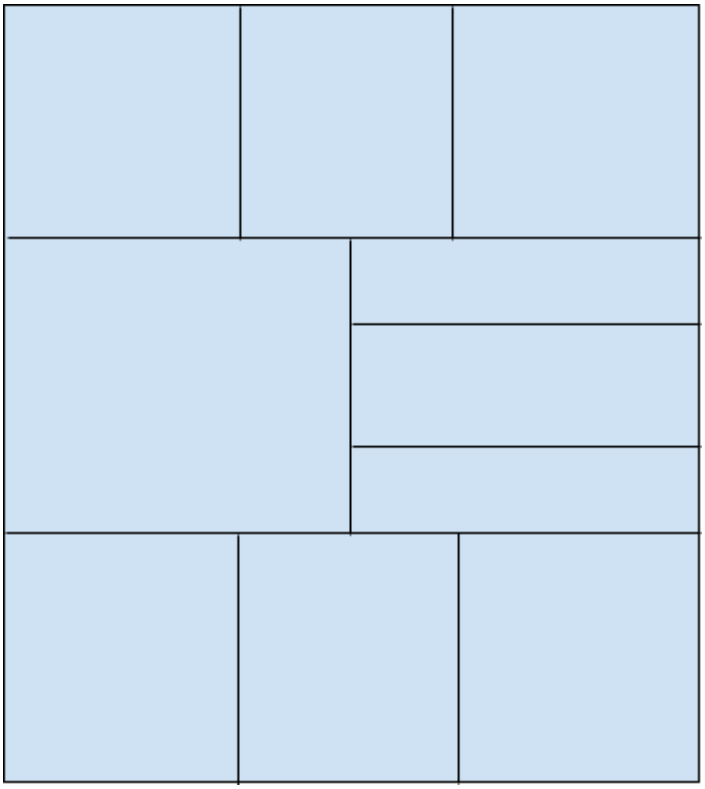
Positions:

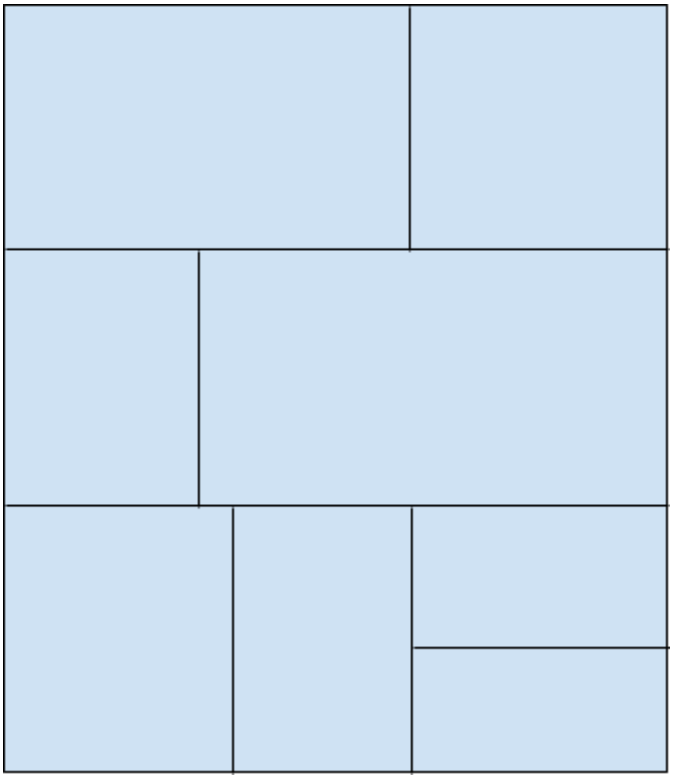


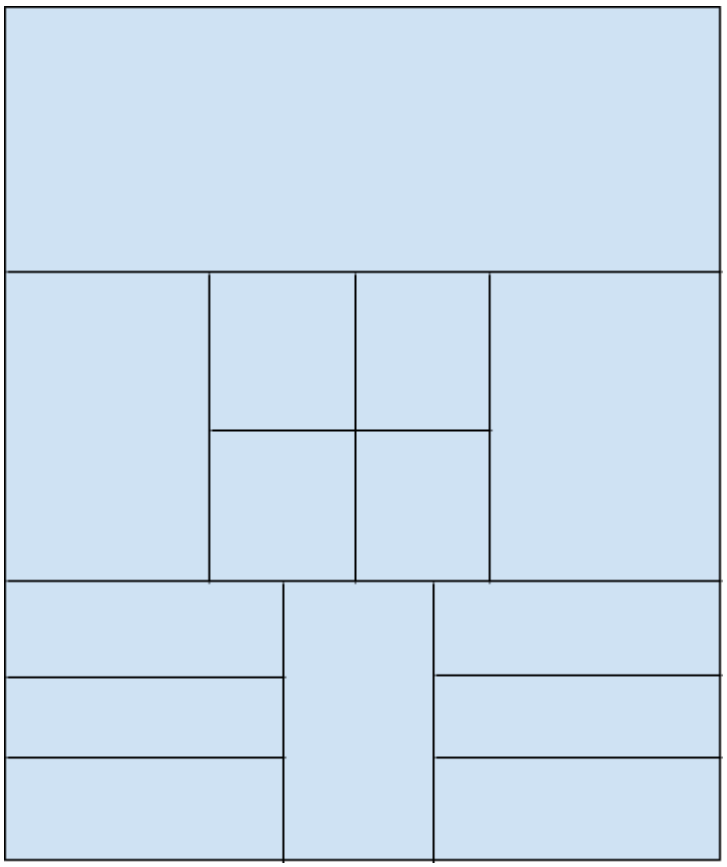



```
<div class = “box1”>box1</div>
<div class = “box2”>box2</div>

<style>
.box1{
        Width:150px;
        Height:150px;
        Background-color:aqua;
        Left:200px;
        position:absolute;/relative
}
.box2{
        Width:150px;
        Height:150px;
        Background-color:brown;
        Top:-50px;
        Position:absolute;
}
```

Assignment 4:







Animations:

@keyFrame name
0% - 100%

Transactions:

<h1>...</h1>
<div

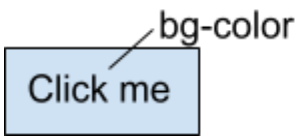
<style>
H1{
Margin-top: 100px;
Transform: rotate(-40deg);
Text-align: center;
Animation-name: mytext;
Animation-duration: 5s;
}
@keyframes mytext{
0%{
Transform: translatex(-500px);
Color: red;
Transform: rotate(30deg);
}
25%{
Transform: translatex(-200px);

```
Color: blue;
}
50%{
Transform: translatex(500px);
Color: chartreuse;
}
75%{
Transform: translatex(900px);
Color: chocolate;
}
100%{
Transform: translate(0);
Color: cornflowerblue;
}
}
.box1{
Width: 120px;
Height: 120px;
Background-color: red;
Animation-name: mytext;
Animation-duration: 5s;
}
```

Hover:

```
<h1>...</h1>

<style>
H1{
Text-align: center;
}
H1:hover{
Color:red;
}
```

Fontawesome - icons
Icons
Phone



HTML

```
<button class=”btn”>Click me</button>
```

CSS

```
.btn{
W:
```

H:
Bg:
Color:
Border:
}

Bootstrap:

<button class=”btn”>Click me</button>

<style>
.btn{
Width: 200px;
Height: 150px;
Background-color: brown;
Color: white;
Font-size: 40px;
Border-radius: 20px;
Border: none;
}

Bootstrap
Responsive - Any device compatibility
Inclue via CDN
Copy & place in header - bootstrap.min.css & bootstrap.bundle.min.js

<button class=”btn btn-primary m-4”>Click me</button>
...

<style>
.btn{
Width: 200px;
}

Docs - opensource

Container:

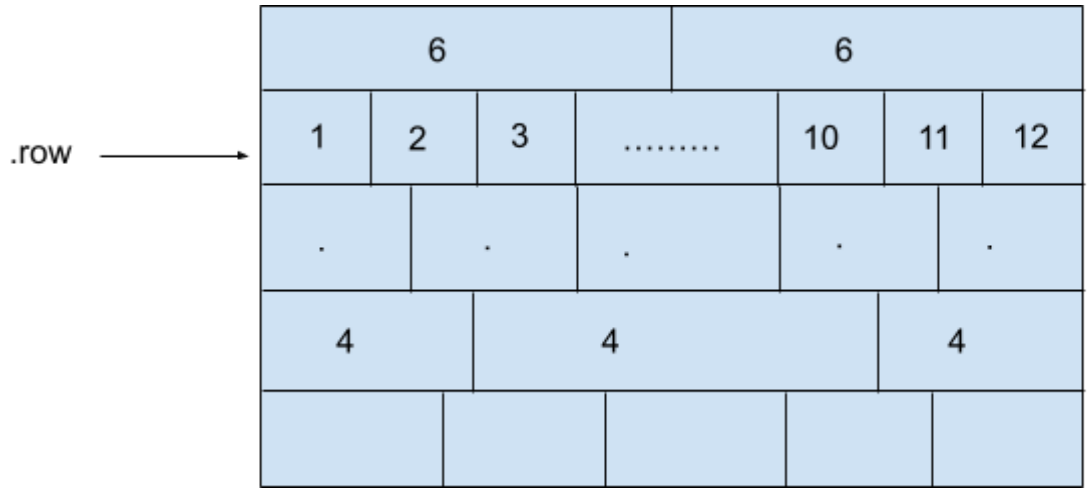
1. Container → Max-width
   .container
2. Container-fluid → Full width
   .container-fluid

<h1>...</h1>

<h1>...</h1>

<h1>...</h1>

Grid: Combination of rows & columns



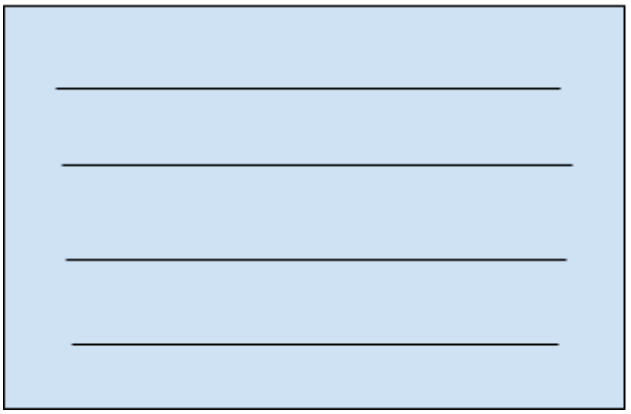
<div class=”col blue”>1
.
.
.
<div class=”col red”>12

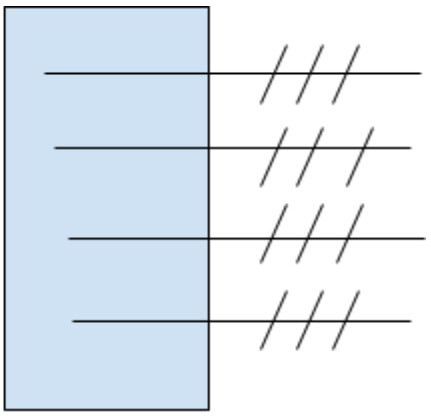
<style>
.red{
Background-color: red;
}
.blue{
Background-color:blue;
}

Inspect → responsive
...
<div class=”col-8 red”>1
<div class=”col-4 blue”>2

Assignment 5:
Bootstrap grid with previous assignment

Breakpoints:





Col-sm → 576px
Col-md → 768px
Col-lg → 992px
Col-xl → 1200px
Col-xxl → 1400px

<head>
<link…
<script… → bootstrap
</head>
<body>

<h1>...</h1>
...
... → 1 - 6
... → 1- 6
...
...
Lorem ipsum

Bootstrap document → Content → Typography

Text & Background colors:

class=”txt”

1. Primary
2. Success
3. Info
4. Warning
5. Danger

6. Secondary
7. White
8. Dark
9. light

<p class=”text-primary”>....
<p class=”text-info”>...
<p class=”text-warning”>....
<p class=”text-dark”>....
<p class=”text-success”>...
<p class=”text-secondary”>...
<p class=”text-light”>....

Text-primary-30

<p class=”text-primary bg-danger”>
“Text-info bg-primary”
“Text-warning bg-info”
“Text-dark bg-light”
“Text-success bg-

Table:

<table class=”table”>
<tr>
<th>ID</th>
<th>Name</th>
<th>Location</th>
</tr>
<tr>
<td>101</td>
<td>Vidya</td>
<td>rjy</td>
</tr>

“Table table - striped”
-bordered
-hover
-dark table-striped table-hover
-borderless
-primary table-striped table-hover
-sm
“Table-responsive”

Content → Tables

Images:

```
<div class=”container-fluid”>
<div class=”row”>
<div class=”col-md”>
<img src=”img/3.jpg” alt=”” class=”rounded”>
</div>
</div>
</div>
```

Img-thumbnail
Rounded-circle

Content → Images

Jumbotrons:

```
<div class=”container-fluid”>
<div class=”row”>
<div class=”col-md”>
<div class=”bg-primary text-danger mt-4 p-5 rounded
<h1>...</h1>
<p>...</p>
</div>
</div>
</div>
```

Alerts:

```
<div class=”container-fluid”>
<div class=”row”>
<div class=”col-md”>
<div class=”alert alert-info alert-dismissable”>
<p>....</p>
<strong>Click me!</strong> —> <a href=”#” class=”alert-link”>Read more!</a>
<button type=”button” class=”btn-close” dat-bs-dismiss=”alert”></button>
</div>
```

Button:

```
<div class=”container-fluid”>
.
.
.
<div class=”d-grid”>
<button class=”btn btn-outline-primary btn-lg btn-block”>Click</button>
```

Components → button

Cards:

```
<div class=”row”>
```

Navbar

Slide
<div class=”row mt-5”>
Card
.
.
.

Forms:

<div class=11 mt-4 mb-4”>
<label for=”fname” class=”form-label”>FirstName</label>
<input type=”text” id=”fname” class=”form-control”>

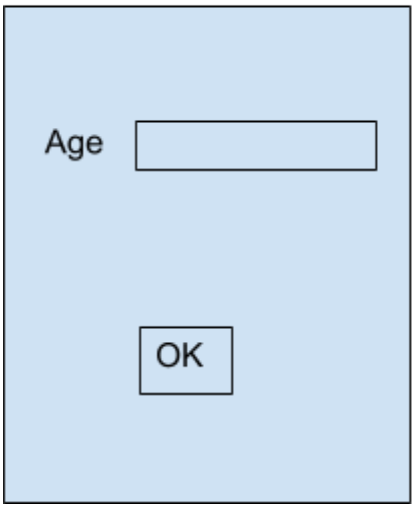
class=”mt-3 mb-3”>
<label for=”lname”>Lastname</label>
<input type=”text” id=”lname” placeholder=”Enter lastname” class=”form-control”>

Forms → Form control

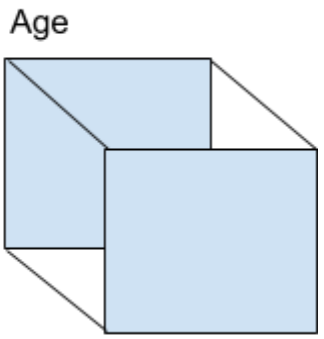
Assignment 6:
Tables & Forms in html billing using bootstrap
Flipkart sign up form
Pixabay → Cards
Royal Enfield → Clone
gvkinfradevelopers.com → Clone

JavaScript:

- Low level
    - Assembly Language (1,0)
- High level
    - General purpose → (C, C++, Java, Python, .net, C#, R)
    - Scripted → JS
    - Special → Android
    - OOPS → Java, Python



Keyword/database
identifier
Int age = 32;
Age = 32;
Int = 32



Compiler - File to file translate
Interpreter - Line to line translate

- Editor
- Browser

<h1 id = “test”>...</h1>

<script>
document.getElementById(‘test’).innerHTML = “Welcome to js”;
</script>

- Head
  <head>
  <script>
  Document.get
  </script>
  </head>
- Body
- External
  myscript.js

Function msg()
{
document.write(“hello”);
}

<script src = “myscript.js”></script>

<input type = “submit” value = “click” onclick = “msg()”>

<script>
Function msg()
{

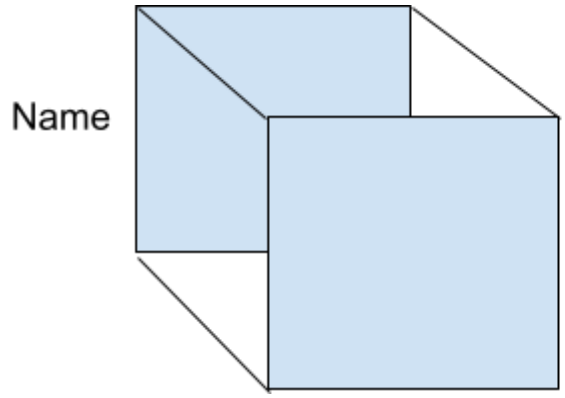
```
document.write("Welcome to js….!");
}
</script>
```

Memory/Storage:

- Primary
    - RAM
    - ROM
- Secondary
    - HDD
    - CD
    - Floppy Disk

Containers:

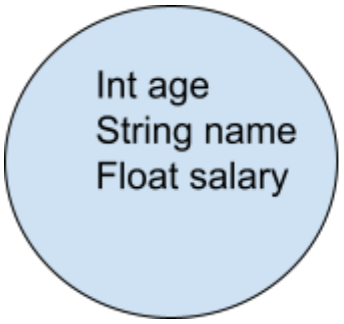
- Variable



String name = "vidya";
Int x = 100;

- Arrays

- Objects



1. Var - rewrite
2. Let - rewrite
3. Const

```
Var x = 100;
document.write(x); 100
X = 1000;

Let x = 1000;
X = 2000;

Const dob = 1993;
Let year = 2024;
Let age = year - dob;

<script>
var x = 100;
Var y = 200;
Var z = x+y;
document.write(z);
</script>

var/let/const var x = 100;
document.write(x+"<br>");
X = 1000;
document.write(x);
```

Inspect → Console

Operator: Symbol

1. **Arithmetic:**

+	10+10=20	
-	10-10=0	
*	10*10=100	
/	10/10=1	Quotient
%	10%10=0	Remainder

```
<script>
Let x = 100;
Let y = 200;
document.write("Sum of two numbers"+(x+y)+"<br>");
document.write("Sub of two numbers"+(x-y)+"<br>");
document.write("Mul of two numbers"+(x*y)+"<br>");
document.write("Div of two numbers"+(x/y)+"<br>");
document.write("Mod of two numbers"+(x%y)+"<br>");
```

Assignment 7:

- Power bill system
  Units = 100 → 2rs
  Price = 2rs
  Total =

- Super market billing system
- Marks
  S1 s2 s3 s4 s5 s6
  Total marks
  Average

2. **Comparison / Relational operators** - Boolean values T/F
   <script>
   Let x = 10;
   Let y = 20;
   document.write(x==y);
   document.wirte(x<=y);
   document.wirte(x>=y);
   document.wirte(x!=y);
   document.wirte(x<y);
   document.wirte(x>y);
   </script>
3. **Bitwise Operators**
   AND - &
   OR - |
   XOR - ^
   NOT - ~

A = 20
B = 10

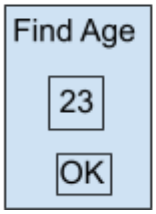
2 | 20 - 0	16	8	4	2	1	0	
2 | 10 - 0	2	2	2	2	2	2	
2 | 5 - 0	1	0	1	0	0	0	
2 | 2 - 1							
1 - 0							

(10 == 30 ^ 20 == 33)
F F = F

1	0	1	0	0		
0	1	0	1	0		
0	0	0	0	0	-	&(0)
1	1	1	1	0	-	1(36)

4. **Logical operators**
   AND - &&
   OR - ||

Age	G1	G2
	1-15	50-100
	5000/-	5000/-



(age >= 1 && age <= 15) || (age >= 50 && age <= 100)
<script>

```
Let age = 10;
if((age >= 1 && age <= 15)||(age >= 50 && age <= 100))
{
        document.write("got 5000");
}
Else
{
        document.write("not get");
}
</script>
```

20 > 45 — F

60 <= 75 — T

AND

1	1	1
1	0	0
0	1	0
0	0	0

OR

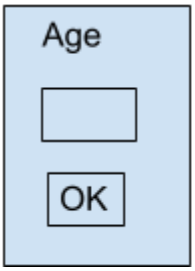
1	1	1
1	0	1
0	1	1
0	0	0

5. Assignment Operator

X = 100 20 =
X = 100 + 20 +=
X = x + 20 -=
X = 120 *=
X += 20 /=
%=

6. Conditional Operator

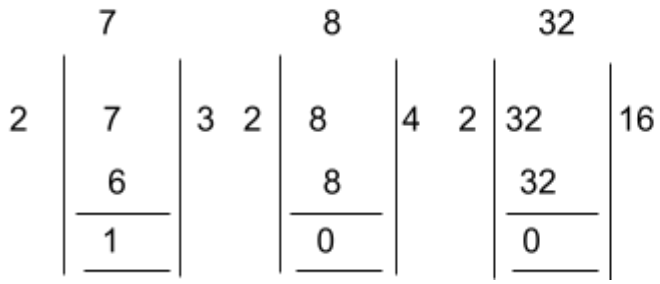
Application



Driving Licence
18
18-60

Age >= 18 && age <= 60

```
<script>
Let age = 18;
if(age >= 18 && age <= 60)
{
        document.write("Eligible for DL");
}
Else
{
        document.write("Not eligible for DL");
}
```




```
Let n = 5;
if(n%2==0)
{
        document.write("Even");
}
Else
{
        document.write("Odd");
}
Marks = 35-100
if(m >= 35 && m <= 100)
{
}
```

If else Ladder:

```
<script>
Let m = 23;
if(m >= 35 && m <= 55)
{
        document.write("C Grade");
}
Else if(m >= 55 && m < 75)
{
        Document.wirte("B Grade");
}
Else if(m >= 75 && m <= 100)
{
        document.write("A Grade");
}
```

```
Else
{
        document.write("Not Valid");
}
</script>
```

Switch(Expression) – int / String

```
{
        Case 1: _________
                Break;
        Case 2: _________
        Case 3: _________
        .
        .
        .
        Default: _________
}
<script>
Var ch = "a";
Var r;
switch(ch)
{
        Case "a":
                R = "a";
                Break;
        Case "b":
                R = "b";
                Break;
        Default:
                R = "not valid";
}
document.write(r);
```

Assignment 8:

1. Speed Test

20 - 45 km	46 - 74 km	75 - 120 km
Normal	Moderate	Danger

2. Billing System

Product-id:
Product-name:
Price:
Quantity:
OP
Billing System
P_id:
P_name:
Price:
Quantity:

Total:
GST:
Discount:
Grand total:

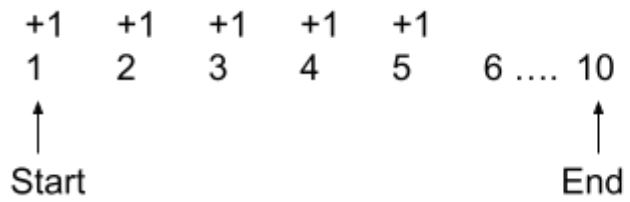
3. Condition
    a. T = 1000 - 20000
       GST 12%
       Offer 5%
    b. T = 21000 - 39000
       GST 13%
       Offer 8%
    c. T = 40000 - 100000
       GST 18%
       Offer 10%

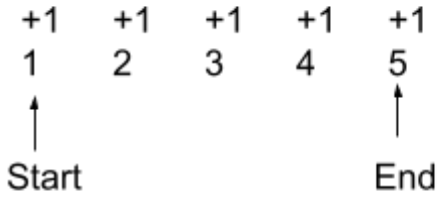
I/P:
Price
Quantity
T = p*q
O/P:
Price
Quantity
Total
GST
Offer
Grand total

Iterations/Loops:

1. For loop
2. While loop
3. Do-while loop

Hello
document.write("Hello");
document.write("Hello");






```
        1       2       4
for(int i=1;i <= 5;i++)
{
        3
}
```

for(Expression 1;Expression 2;Expression 3)
{
}

Exp-1
Initialization
Int i = 1

Exp-2
Condition
I <= 5

Exp-3
increment/decrement
i++/i–

For loop:

```
<script>
for(let i=1;i <= 5;i++)
{
        document.write(i+”hello”);
}
</script>
```

-1 -1
10 9 8 1
I = 10;i >= 1;i–

While	**Do while**
Pre Condition	Post Condition

While(Exp)
{
Statements
}
	Do{
	Statements
	}while(Exp);

While:

```
<script>
Let i = 1;
while(i <= 10)
{
        document.write(“hi”);
        I++;
}
</script>
```

Do while:

```
<script>
Let i = 1;
Do{
        document.write(i);
        I++;
}while(i <= 10);
```

```
1 * 5 = 5
2 * 5 = 10
3 * 5 = 15
4 * 5 = 20
.
.
.
10 * 5 = 50
```

```
for(let i = 1;i <= 10;i++)
{
        d.w(i+”*”+”5”+”=”+(i*5));
}
```

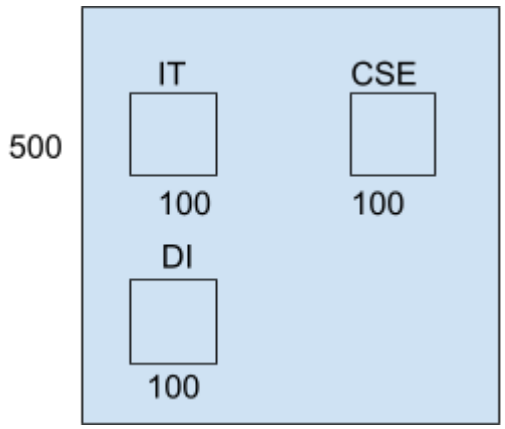
Sum of N numbers:

```
N = 5
1+2+3+4+5=15
Let sum = 0;
for(let i = 1;i <= n;i++)
{
        Sum += i;
}
d.w(sum)
```

```
<script>
```

```
Let n = 10;
Let i = 1;
Let sum = 0;
while(i <= n)
{
        Sum += i;
        I++;
}
document.write(sum);
</script>

Do{
        document.write(i);
        I++;
}while(i >= n);
```

Functions: Reusability & portability

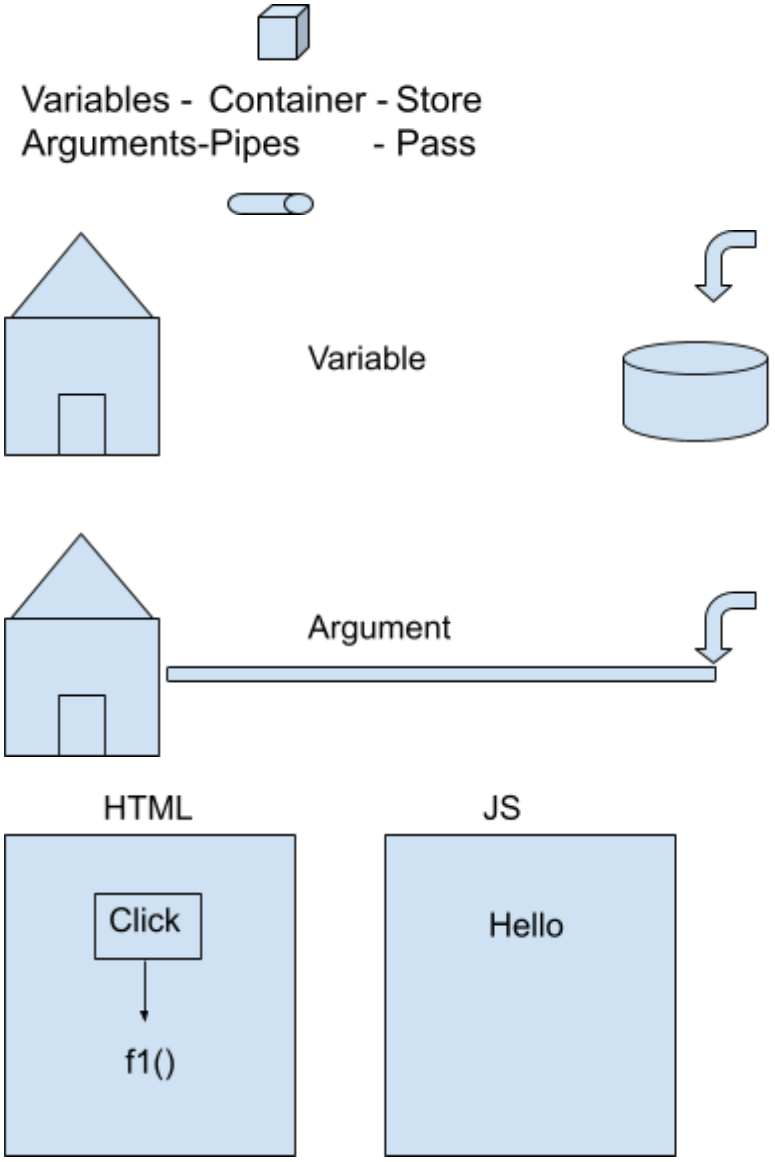



```
d.w(a+b)
d.w(a-b)
d.w(a*b)
d.w(a/b)
d.w(a%b)
```

20 Req → 1 → Registration & Signup

```
Signup()
{
        _____
        _____
        _____
}
Signin()
{
        _____
        _____
        _____
```

}
Function add(arg1, arg2….)
{
	Group of statements
}



Onclick

Function f1()
{
	alert(“Hello”);
}

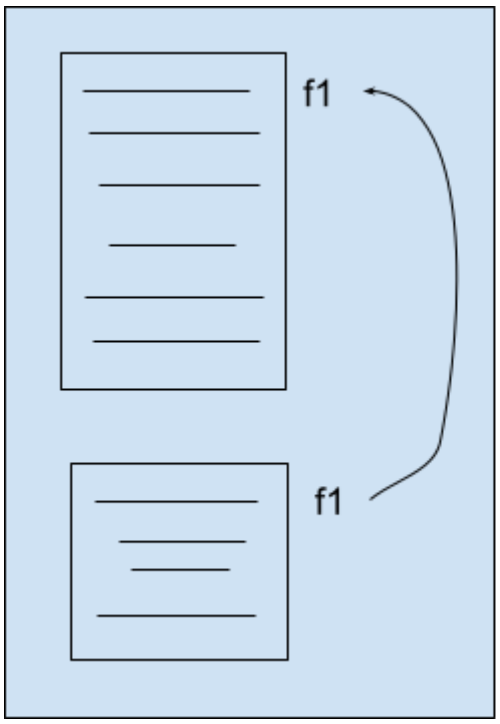
<body>
<input type = “button” value = “click” onclick = “f1()”>
<script>
Function f1()
{

```
        alert("Hello");
}
</script>
</body>
```

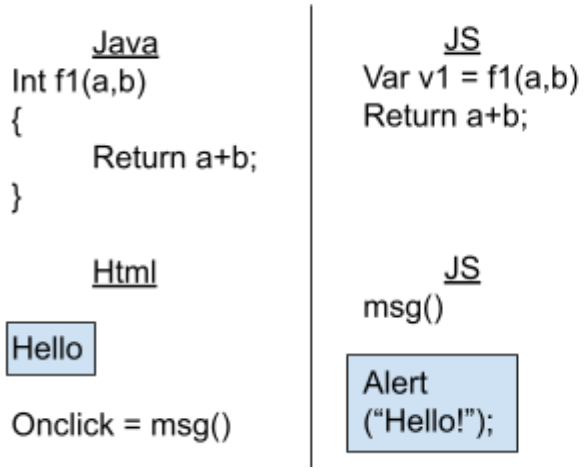
Assignment 9:

1. Factors
   N = 5
2. Factorials
   5!
3. 1-100 Sum of odd numbers

Function:



1. Header / Prototype
2. Body

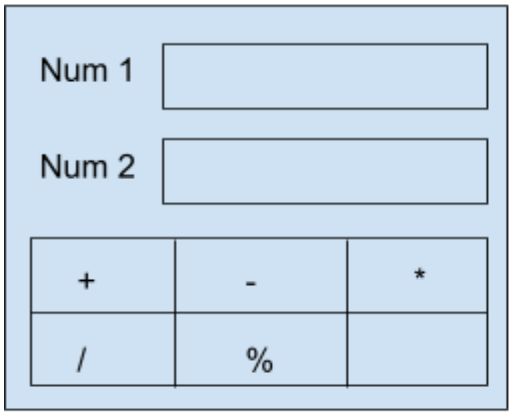



```
<input type = "button" value = "click" onclick = "msg()">
<script>
```

```
Function msg()
{
        alert(“Hello goodmorning!”);
}

Function add(a,b)
{
        alert(a+b);
}

Onclick = add(2,3)

Function msg(a,b)
{
        Return a+b;
}

document.write(msg(10,20));
```

Assignment 10:

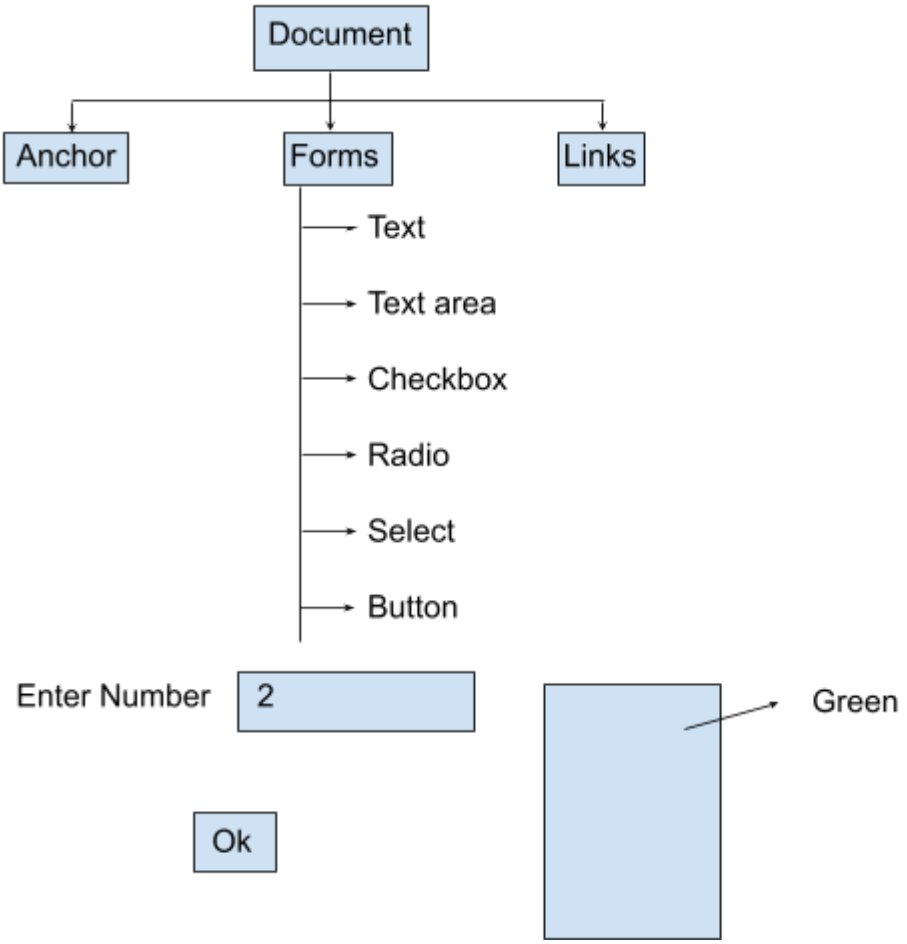
HTML



JS

```
<span id = “r1”>Result:</span><br>
<label for = “n1”>Num 1:</label>
<input type = “text” id = “n1” placeholder = “Enter Number - 1”> <br>
<label for = “n1”>Num 2:</label>
<input type = “text” id = “n1” placeholder = “Enter Number - 2”> <br>
<input type = “submit” value = “+” onclick = “add()”>
<script>
Function add()
{
        Let v1 =parseInt( document.getElementById(‘n1’).value);
        Let v2 =parseInt( document.getElementById(‘n2’).value);
        document.write(v1+v2);
        document.querySelector(“#r1”).innerHTML = “Sum of two numbers:” + (v1+v2);
}
```

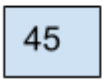
DOM - Document (.) Object Model



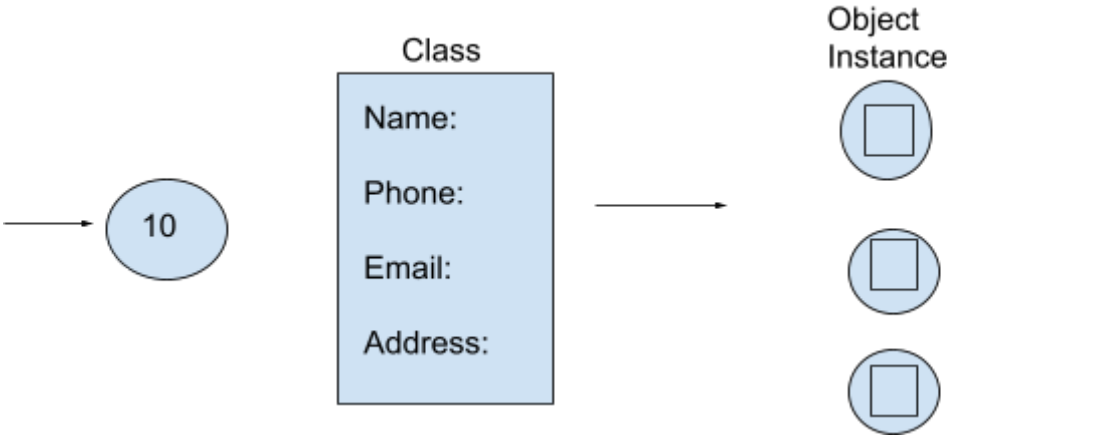

```
Function f1()
{
	Let v = G.E.Id();
	if(v%2 == 0)
	{
	}
	Else
	{
	}
}

<label for = “n”>Enter Number</label>
<input type = “text” id = “n”>
<br>
<input type = “submit” value = “click” onclick = “m1()”>
<script>
Function m1()
{
	Let v = document.querySelector(“#n”).value;
	if(v%2 == 0)
	{
		document.querySelector(“body”).style.backgroundColor = “green”;
```

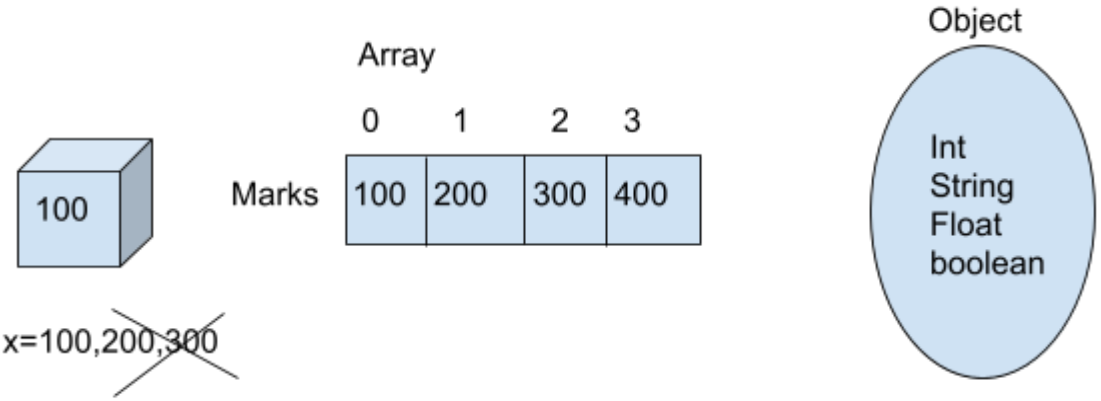
```
        }
        Else
        {
                document.querySelector("body").style.backgroundColor = "red";
        }
}
```

Assignment 11:



Objects:





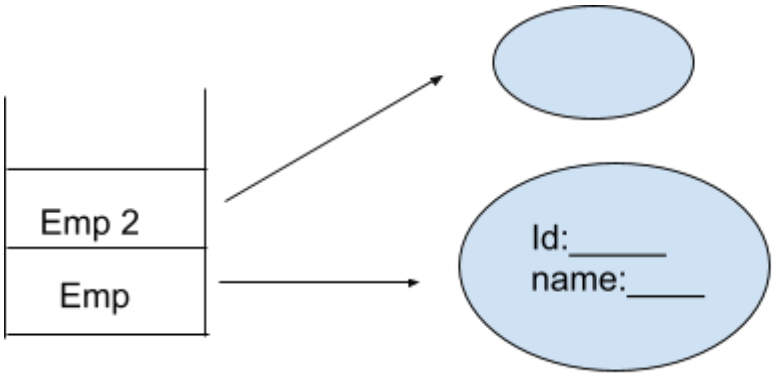
1. Object Literal

2. New Keyword
3. Object constructor

Object Literal:
Let x

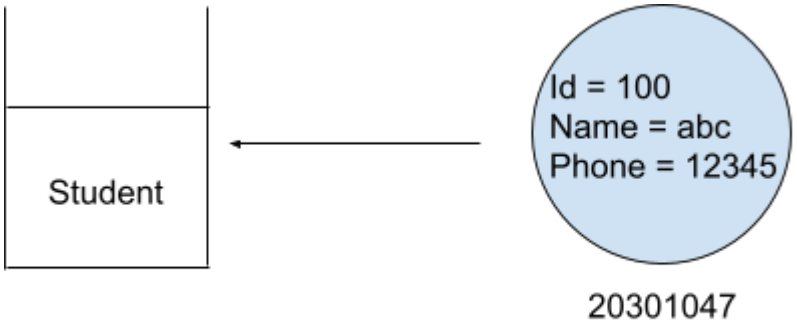
Variable Literal:
Let x = 100;

Emp = {id:102, name:”vidya”}



<script>
emp={id:101,name:”vidya”};
document.write(emp.id+””+emp.name);
emp1={id:44,name=”Suresh”};
document.write(emp1.id+””+emp1.name);
</script>

New keyword:
<script>
Var student = new object();
student.id=100;
student.name=”abc”;
student.phone=12345;
document.write(student.id+””+student.name+””+student.phone);

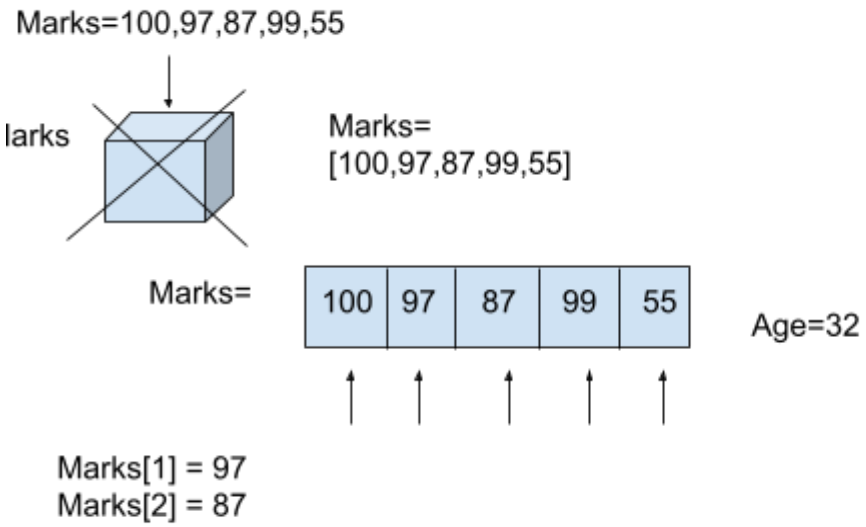


20301047

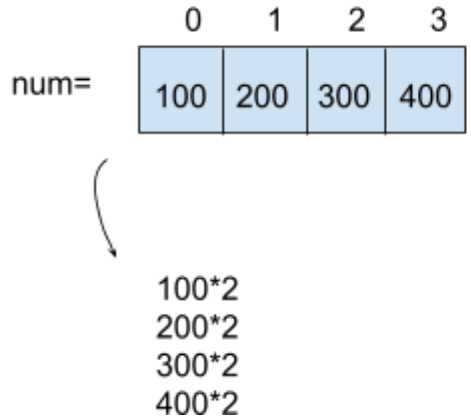
Object Constructor:
<script>
Function emp(id,name,sal)

{
this.id=id;
this.name=name;
this.sal=sal;
}
e=new emp(100,”vaidya”,34000);
document.write(e.id+””+e.name+””+e.sal);
</script>

Arrays:



<script>
Let names = [“abc”,”xyz”,”pqr”];
document.wirte(names);
for(let i=0;i<names.length;i++){
document.write(names[i]+”
”);

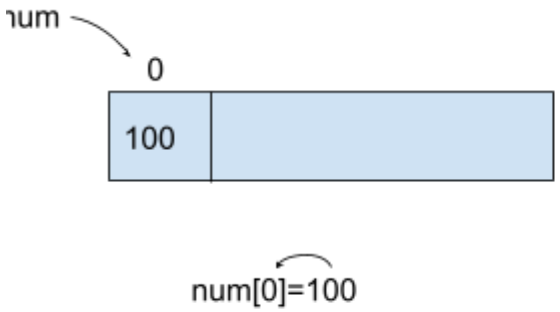


Assignment 12:

1. Let num = [100,200,300,400]
   for(let i = 0;i<num.length;i++)
   {
   Let sum = num[i]*2;

New:

```
<script>
Let num=new Array();
num[0]=100;
num[1]=200;
num[2]=300;
num[3]=400;
for(let i = 0;i<num.length;i++)
{
document.write(num[i]*2);
}
</script>
```

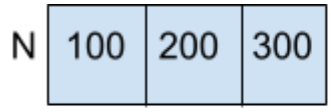


String:

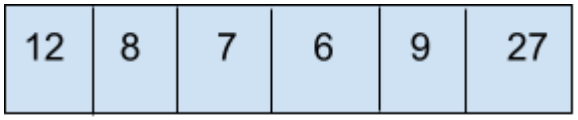
```
Let name=""
Let name=new String("")
d.w(name)
```

Assignment 13:

Sum of N numbers using array



Sum of even:




```
.container{
width:100%;
Height:100vh;
Background-color:aqua;
}
.block-1{
width:100%;
Height:50vh;
Background-color:brown display:flex
}
```

<div class=”box-1”

Jquery:

Jquery→ download
Google cdn
3x snippet → copy
Paste in head tag

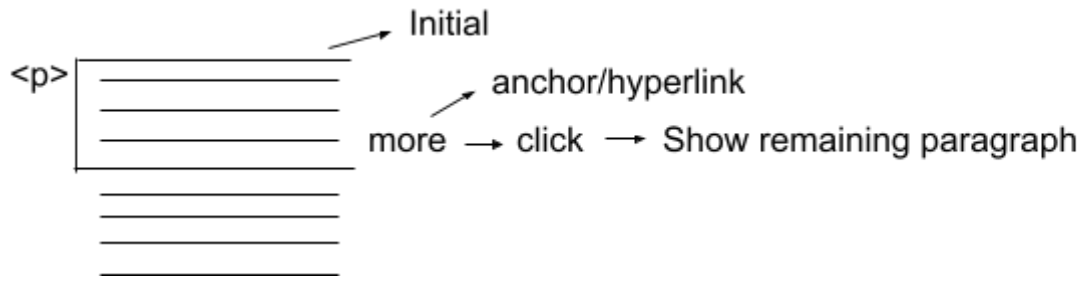
Display:

1. hide()
2. show()
3. toggle()

Hide/show:

```
<style>
P{
Width:400px;
Height:400px;
Display:none;
}

<p>....</p>
<button id=”hide”>hide</button>
<button id=”show”>show</button>
<button id=”toggle”>toggle</button>

<script>
$(document).ready(function(){
$(“#hide”).click(function(){
$(“p”).hide(3000);
$(“#show”).click(function(){
$(“p”).show(1000);
});
$(“#toggle”).click(function(){
$(“p”).toggle(1000);
});
});
```

Assignment 14:



Fade:

```
<style>
.box1{
Width:150px;
Height:150px;
Background-color:blue;
Display:none;
}
</style>
<button id = “fadein”>fade in</button>.box1<button id = “fade out”>fade out</button>
<button id = “fadetoggle”>fade toggle</button>
<script>
$(document).ready(function(){
$(“#fadein”).click(function(){
$(“.box1”).fadein(1000);
});

$(“#fadeout”).click(function(){
$(“.box1”).fadeout(1000);
});
$(“#fadetoggle”).click
```

Slide:

```
#slidetoggle slidetoggle
#click slidedown
#body
Lorem….
….
…..
#up slideup
<style>
#click,#up,#slidetoggle{
Text-align:center;
Background-color:aquamarine;
Border:1px red solid;
Padding:30px;
}
#body{
Text-align:center;
Background-color:chartreuse;
```

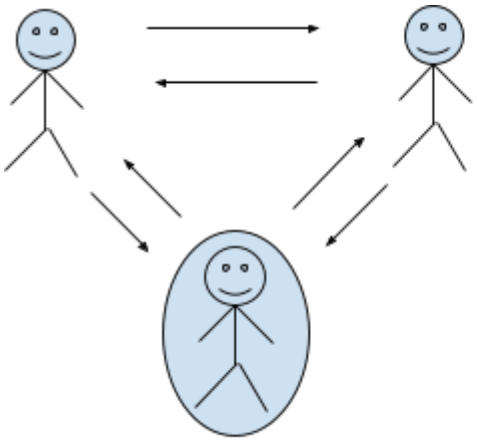
```
Border:1px solid gray;
Display:none;
Padding:30px;
}

<script>
$(document).ready(function(){
$("#click").click(function(){
$("#body").slidedown(1000);
});
$("#up").click(function(){
$("#body").slideup(1000);
});
$("#slidetoggle").click(function(){
$("#body").slideToggle(1000);
});
});
```

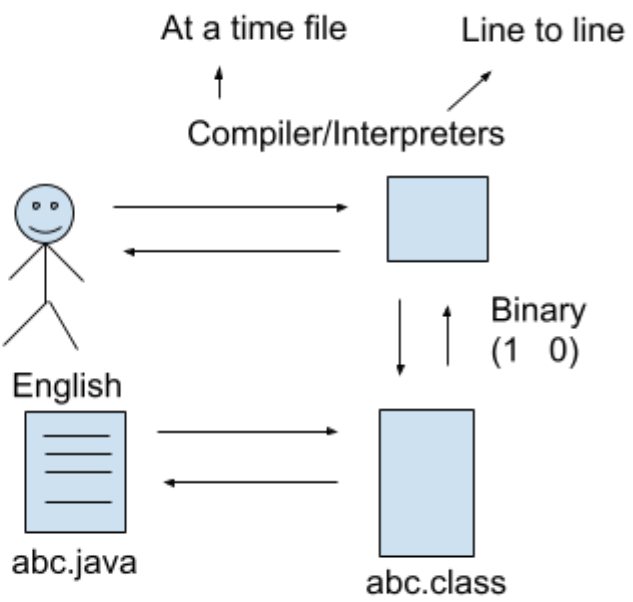
Animate:

```
#click{
Background-color:chocolate;
Padding:20px;
Width:50px;
Height:10px;
}
.box1{
Width:100px;
Height:100px;
Background-color:red;
Position:absolute;
}
<p id = "click">click me</p>
.box1

<script>
$(doucment).ready(function(){
$("#click").click(function(){
$(".box1").animate({left:'350px'});
});
```

Jquery → document

Java





Low level
(1 0)

High level
General purpose

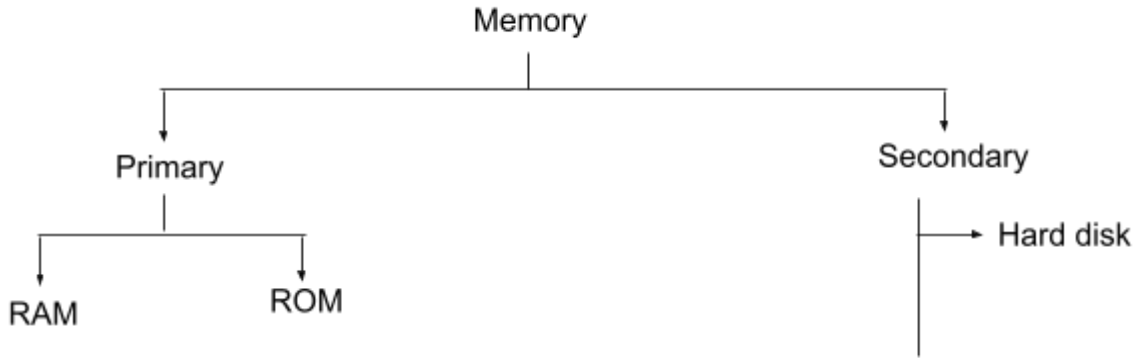
Compiler: c,c++
Compiler and interpreter: Java,python
Interpreter:
Scripted: js,vb
Special: android

1. Int age = 35
2. Age = 35
3. Int = 35

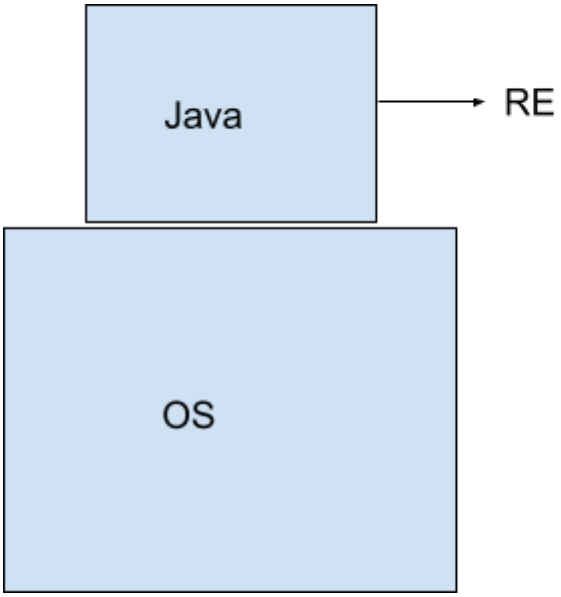
Int → Datatypes/reservewords/keywords
Age → identifiers

```
Int sum()
{
}
```

Class Employee



1. JDK
2. Editor/(Notepad, Notepad++)
3. IDE → Eclipse, Intellij, VS



Class Test
{
Public static void main(String argos[])
{
System.out.println(“Hello”);
}

Class → Keyword
Test → Identifier

Javac Test.java

Operators:

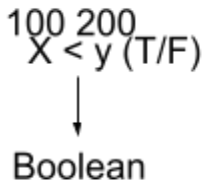
1. Unary operators
    a. Postfix → post increment (Exp++)/ post decrement (Exp - -)
    b. Prefix → pre increment (++Exp)/ pre decrement (--Exp)

X = 100
Int x = 100;
(x++) //100
(++x) //102
(--x) //101
(x--) //101

10+10
-
*
/
%

2. Conditional / Comparison:

<
>
<=
>=
= =
!=



3. Assignment operator

X = 100
X = 100 + 10
X = x + 10
X -= 10
X /= 2
X %= 2

4. Bitwise & Logical Operators

1. Logical
    a. AND
    b. OR
2. Bitwise
    a. AND
    b. OR
    c. NOT

AND

1	1	1
0	1	0
1	0	0
0	0	0

20>10

1

OR

1	1	1
0	1	1
1	0	1
0	0	0

10<5

0

Shift Opeartors:

Left (<<) Right (>>)

N = 20 << 2

2 | 20

2 | 10 – 0

2 | 5 – 0

2 | 2 – 1

1 – 0

Left Shift:

2^6	2^5	2^4	2^3	2^2	2^1	2^0
64	32	16	8	4	2	1
		1	0	1	0	0
1	0	1	0	0	0	0
		80				

20 * 2 * 2
20 * 4 = 80

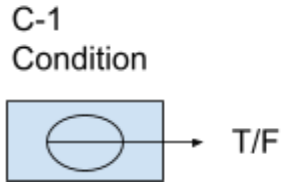
Right Shift:

1 0 1 0 0 0 N = 40 >> 2
1 0 1 0 0 0 10

40/2*2
40/4 = 10

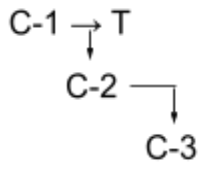
Conditional Statements:

1. If
2. If-Else
3. Else-If
4. Nested If
5. Switch



<
<=
>
>=
==
!=

C-1
C-2
C-3
C-4



IF:

```
if(Condition) T
{
—
—-
—-
}
```

IF-Else

```
if(Condition) T/F
{
        True
}
Else
{
        False
}
```

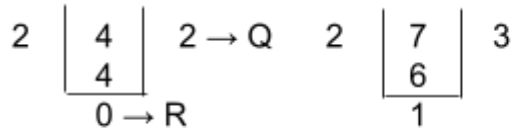
Else-if Ladder:

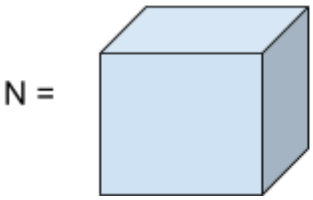
```
if(C-1)
{
}
Else if(C-2)
{
}
Else if(C-3)
{
}
Else
{
}
```

Nested If:

```
if(C-1)
{
        if(C-2)
        {
        }
        Else if(C-3)
        Else
        {
        }
Else
{
}
```

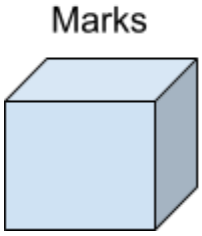
Even/odd:






```
if(n%2==0)
{
}
Else
{
}
```

Mark = 35




```
if(m>= 35)
{
}
Else
{
}
```

```
if(marks>=35 && m<= 100)
{
}
Else
{
}
```

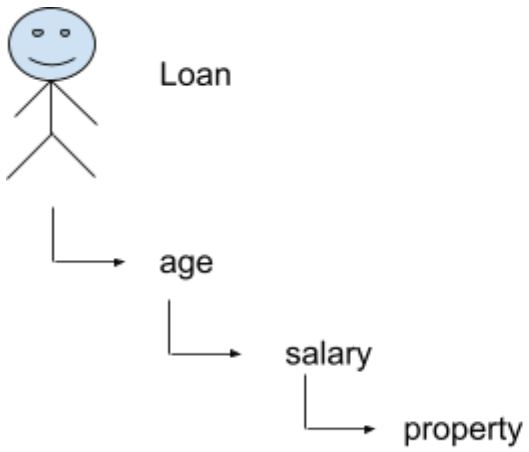
5000
Age-1 1 - 10
Age-2 50 - 100

```
if(age>=1 && age<=10 || age>=50 && age<=100)
```

Marks = (35-100) → P

35-54 = C
55-74 = B
75-100 = A

```
if(35-54)
{
}
Else if(55-75)
{
}
Else if(75-100)
{
}
Else
{
}
```



Age = 18 - 55
Sal = 18000 - 100000
Property = 50L - 1Cr

```
if(age>=18 && age<=55)
{
        if(sal>=18000 && sal<=100000)
        {
                if()
                {
                }
        }
}
```

Switch:

```
switch(Expression) → Integer, Characters & Strings
{
        Case 1:
```

```
        Break; (optional)
    Case 2:
    Case 3:
    Default:
}
```

Assignment-15:

1. Even or odd
2. Marks
3. Driving Liscense
4. Super Market Billing

P:3000
R:2
Total:6000
GST(18%)
offer(5%)
5000-15000
5000-14999 (5%)
15000-24999 (8%)
25000-49999 (10%)
GT

Loops:

1. For
2. While
3. Do-while

For:

```
For(exp 1;exp 2;exp 3)
{
}
```

Exp 1 - Initialisation
Exp 2 - Condition
Exp 3 - Increment / decrement

```
for(i=5;i>=1;i–)
{
}
```

While: Pre Condition

```
Exp 1 (Initialisation)
While (exp 2) (condition)
{
    Exp 3 (Iteration)
}
```

Do while: Post Condition

```
Exp 1 (Initialisation)
Do
{
        Exp 3 (Increment)
}
while(exp 2); (condition)
```

```
N=5
Sum = 0;
for(i=1;i<=5;i++)
{
        sum+=i;
}
```

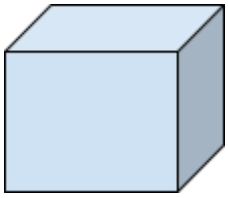
OOP's:

1. Object
2. Class
3. Inheritance
4. Polymorphism
5. Abstract
6. Interface
7. This → refer to current variable
8. Super
9. Exception
10. Multithreading
11. I/O
12. JDBC

Class:

Types of Container

1. **Variable**



Int age = 23;
Marks = 23,32,47,57
Mark = 23

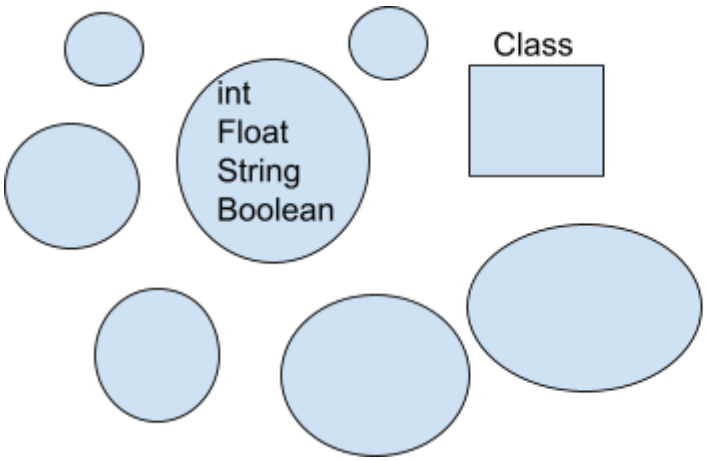
2. **Arrays**

Int marks[5] =

0	1	2	3	4
23	32	47	57	67

Marks[0] = 23,4.5,"ABC"

3. **Object -** Copy of a class



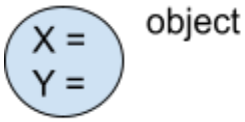
Class is a template contains Variables, Methods and Constructors

Variables:

1. Instance (non Static)
2. Static

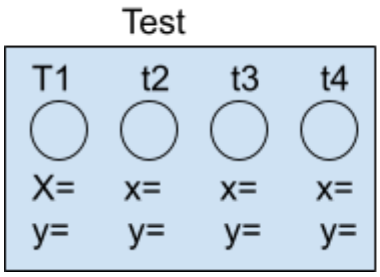
Class	Object
Memory: one time (When class loads)	Memory: Everytime object is created
Static keyword	
Call with class name	Call with object reference
Values are shareable	Values are not sharable

```
Class Test
{
        Static int x = 100;
}

Int x;
Int y;
```

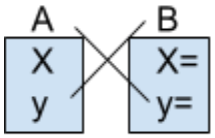


How to initialize the instance variables?

1. Within the class



2. Other class



3. Constructor
4. Get/Set

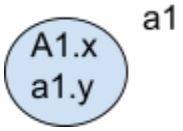
```
Class A
{
        Public static void main()
        {
        }
}
```

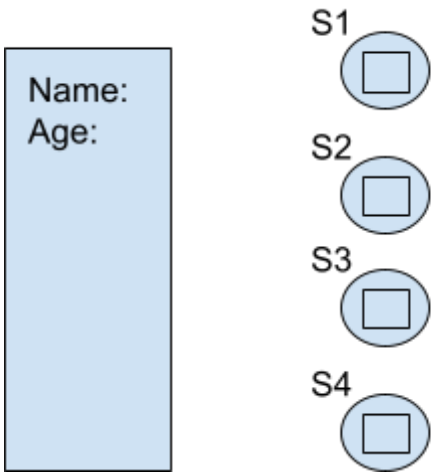
Class without main method:
POJO - Plain Old Java Object

```
Class A
{
        Int x;
        Int y;
}
```

A a1 = new A();
A1.x = 100;
A1.y = 200;
A a2 = new A();
A2.x = 1000
A2.y = 1000

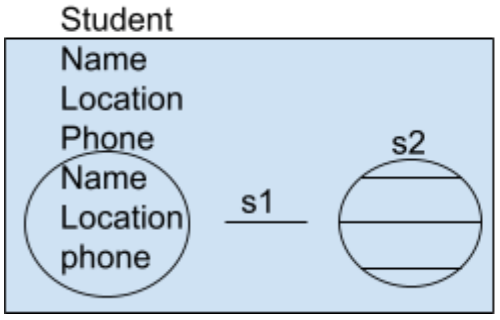
New → Dynamic memory allocation
A() → Constructor





Within the class:

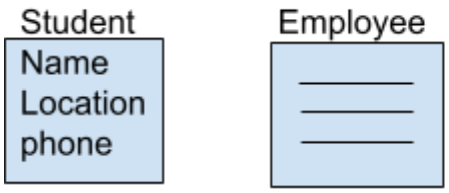
```
Class student
{
        Student name;
        String location;
        String phone;
        Public static void main(String args[])
        {
                Student s1 = new Student();
                s1.name = “ABC”;
                S1.location = “RJY”;
                S1.phone = “123”;
        }
}
```



Other class:

```
Class Student
{
        String name;
        String location;
        String phone;
}

Class Test
{
        PSVM()
        {
```

```
                Student s1 = new Student();
                s1.name = _________
                S1.location = _________
                S1.phone = _______
        }
}
```



Test

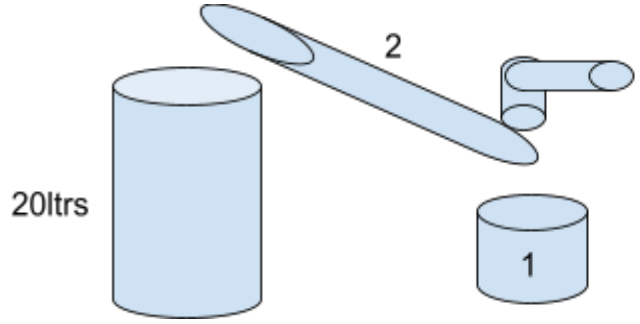
```
Class student
{
        String name;
        String phone;
        String location;
        Public static void main (String args[])
        {
                Student s1 = new Student();
                s1.name = “abc”;
                S1.phone = “123”;
                S1.location = “rjy”;
                Student s2 = new Student();
                s2.name = “xyz”;
                S2.phone = “345;
                S2.location = “hyd”
                System.out.println(s1.name+””+s1.location+””+s1.phone);
                System.out.println(s2.name+””+s2.location+””+s2.phone);
        }
}
```

```
Javac Student.java
Java student
```

```
Class Test123
{
        Public static void main(String args[])
        {
                Student s1 = new Student();
                s1.name = “vidya”;
                S1.location = “vsp”;
                S1.phone = “23456”;
```

```
                System.out.println(s1.name+””+s1.location+””+s1.phone);
        }
}
```

Javac Test123.java



Constructors:

1. Default
2. Parameterized

Int = 0
String = null

Default constructor:

```
Student s1 = new Student();
System.out.println(s1.name+””+s1.location+””+s1.phone);

Class student
{
        String name;
        String phone;
        String location;
        student(string name, string phone, string location)
        {
                This.name = name;
                This.phone = phone;
                This.location = location;
        }
}
Class Test123
{
        PSVM()
        {
                Student s1 = new Student(“suresh”,”hyd”,”12345”);
                System.out.println(s1.name+””+s1.location+””+s1.phone);
        }
}
```

Assignment-16:

Employee: id, name, location, designation → class
Customer: id, name, phone, email, location → class

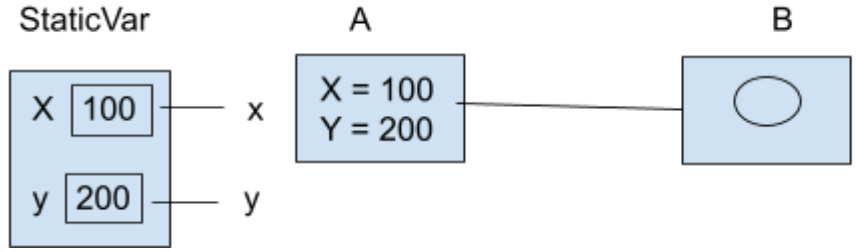
V → Instance - Obj & Static - Data Sharable
M
C

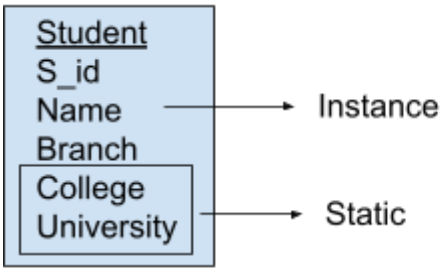
4 Ways: Initialize instance Variables

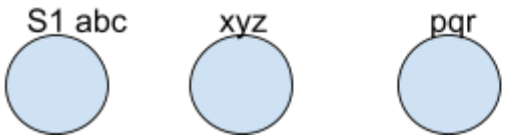
- With in the class
- Other class
- Constructor
    - Default
    - Parameterized
- Using Setters & Getters methods
  set() → x = 100 → get()

```
Class Customer
{
        Int c_id;
        String c_name;
        Public void setId(int c_id)
        {
                This.cid = c_id;
        }
        Public void setName(String c_name)
        {
                This.c_name = c_name;
        }
        Public int getId()
        {
                Return c_id;
        }
        Public String getName()
        {
                Return c_name;
        }
        Public static void main(String args[])
        {
                Customer obj = new Customer();
                obj.setId(101);
                obj.setName(“vidya”);
                System.out.println(obj.getId()+” “+obj.getname());
                System.out.println(StaticVar.x+” “+StaticVar.y);
        }
}

Class StaticVar
```

```
{
        Static int x = 100;
        Static int y = 200;
        Public static void main(String args[])
        {
                System.out.println(x+" "+y);
        }
}
```







Method:

- Instance
- Static
- Factory → return obj

```
Public void function name(list of Args)
{
        Group of statements
}
```

Public → Access specifier/modifier
Void → return type
Public void function name(list of Args) → header/prototype
Function name(list of Args) → signature

1. No return type no argument
   ```
   Public void f1()
   {
   ```

```
        s.o.p("Hi");
    }
```

2. No return type with Arg

```
    Public void f2(int x)
    {
        s.o.p(x)
    }
```

3. With return type No Args

```
    Public int f3()
    {
        Return 10;
    }
```

4. With return type with Arguments

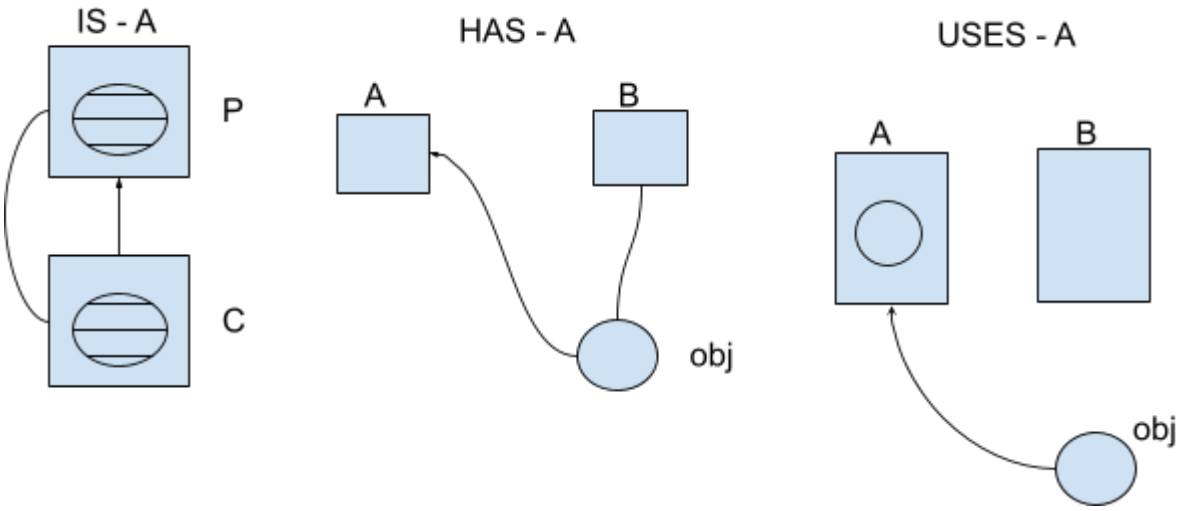
```
    Public String f4(String name)
    {
        Return name;
    }
```

```
Class MethodsExample1
{
    Public void f1()
    {
        System.out.println("Hello");
    }
    Public int f2()
    {
        Return 10;
    }
    Public void f3(int y)
    {
        System.out.println(y);
    }
    Public String f4(String name)
    {
        Return name;
    }
    Public static void main(String args[])
    {
        MethodExample1 obj1 = new MethodExample1();
        obj1.f1();
        Int v1 = obj1.f2();
        System.out.println(v1);
        obj1.f3(100);
        String v2 = obj1.f4("vidya");
        System.out.println(v2);
    }

Class StaticMethod2
```

```
{
        Public static int f1()
        {
                Return 10;
        }
        Public static void f2()
        {
                System.out.println(“Hello”);
        }
        Public static void main(String args[])
        {
                System.out.println(f1());
                f2();
        }
}
```

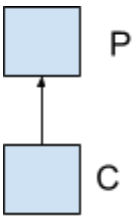
Types of Relationship:

- IS - A
- HAS - A
- USES - A

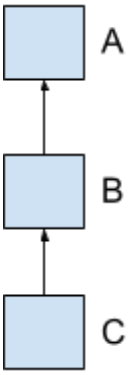



```
Class A
{
        Parent
}
Class B extends A
{
        Child
}
```

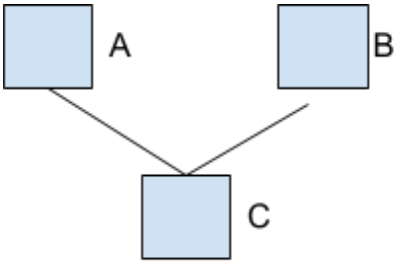
- Single



- Multilevel



- Hierarchical



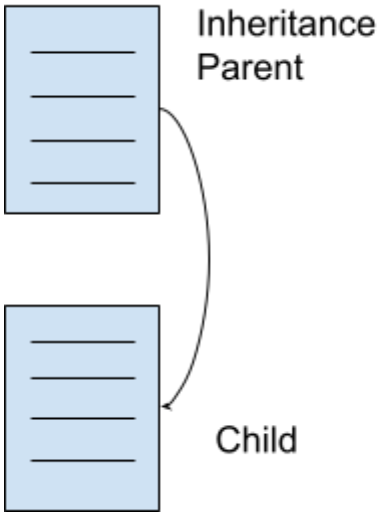
- Multiple
- Hybrid

Assignment 17:

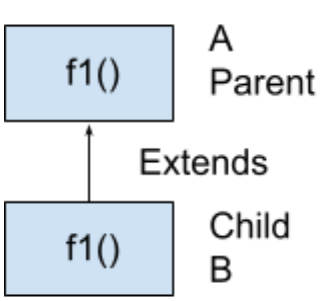
4 types → Instance & Static

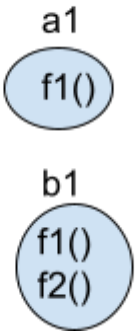
Arithmetic Operations

IS-A:



Single Inheritance:



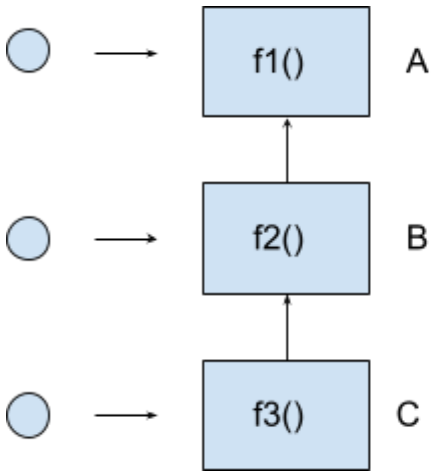



```
A a1 = new A();
a1.f1();
a1.f2();

B b1 = new B();
b1.f1();
b2.f2();

Class A
{
        Public void f1()
        {
                System.out.println("Parent class f1 method");
        }
}
Class B extends A
{
        Public void f2()
        {
                System.out.println("Child class f2 method");
        }
}
```

```
Class TestInh
{
	Public static void main(String args[])
	{
		//A a1 = new A();
		//a1.f1();
		//a1.f2();
		B b1 = new B();
		b1.f1();
		b1.f2();
	}
}
```

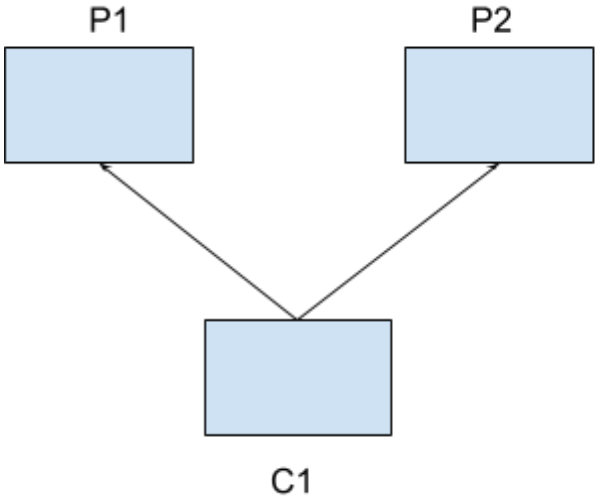
Javac TestInh.java
Java TestInh

Multilevel Inheritance:

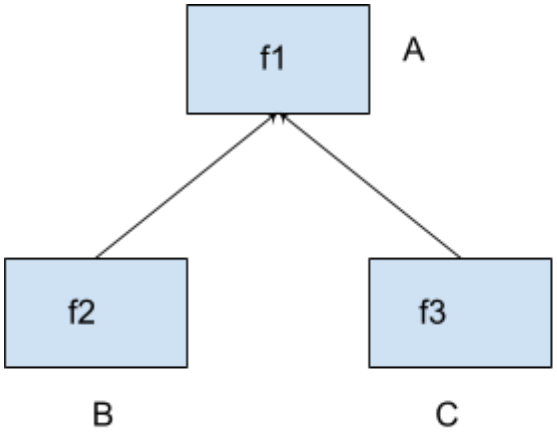



```
Class C extends B
{
	Public void f3()
	{
		System.out.println("Child class f3 method");
	}
}
Class TestInh
{
	Public static void main(String args[])
	{
		C c1 = new C();
		c1.f1();
		c1.f2();
		c1.f3();
	}
}
```

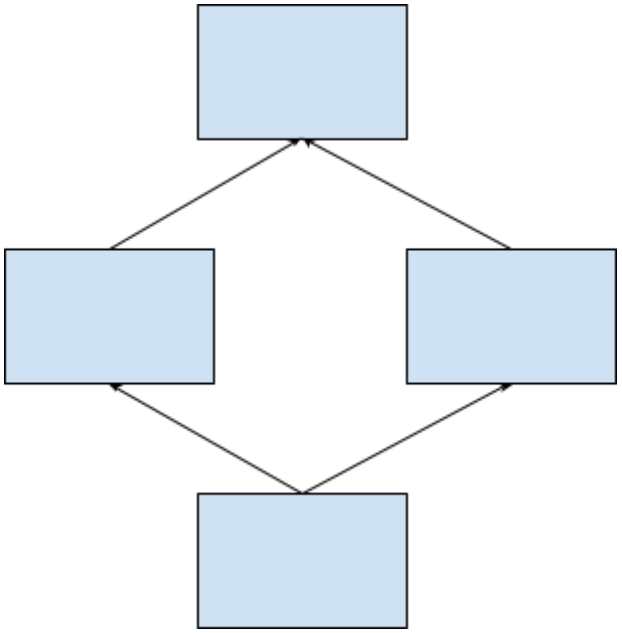
Hierarchical:



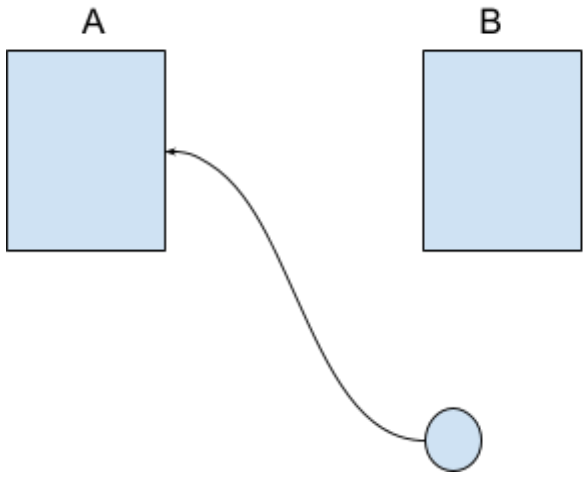
Multiple:



Hybrid:



HAS-A:




```
Class Address
{
        String location;
        String state;
        String country;
        Address(String location, String state, String country)
        {
                This.location = location;
                This.state = state;
                This.country = country;
        }
}
Class Employee1
{
        String name;
        String phone;
        Address a1;
        Employee1(String name, String phone, Address a1)
        {
                this.name = name;
                This.phone = phone;
                This.a1 = a1;
        }
        Public void display()
        {
                System.out.println(name+" "+phone);
                System.out.println(a1.location+" "+a1.state+" "+a1.country);
        }
        Public static void main(String args[])
        {
                Address a1 = new Address("rjy","ap","ind");
                Employee1 e1 = new Employee1("vidya","234567",a1);
                e1.display();
        }
}
```

Assignment 18:

Customer & Shipping → Address → Permanent & Current & Shipping