```
List<Fees__c> lstfee= new List<Fees__c>();

lstfee = [select Email__c,Student_Name__c,Name from Fees__c ];

update lstfee;


S2SIntregrationController.updateRecord();



ExampleBatchClass expBatch = new ExampleBatchClass();

database.executebatch(expBatch,10);

//-----------------------------------------------------------------------------------------------------------------------------


//sObject


//inserting data/a row into object through apex code

Shop__c s = new Shop__c(Shop_Name__c = 'Soham Grocery Store',Location__c='New
Panvel',Timing__c='8am to 10pm',Shop_type__c='Food Store');

insert s;


//SOQL query to extract data from object

Shop__c data = [select Shop_Name__c ,Timing__c,Shop_type__c from Shop__c where
Location__c='New Panvel'];

system.debug('Data:'+data);


//apex Collections:

//List with primitive types like int, String ...


//Define a list

List<integer> myList = new List<integer>();
```

```
//Add 47  to the list
myList.add(47);
system.debug('Value Added to List');
myList.add(55);
myList.add(65);
myList.add(75);
system.debug('Current List:'+myList);



//Retrieve element at index 0
Integer valOfIndex = myList.get(0);

//Add integer 1 to the list at index 0
myList.set(0,1);

//Display List
system.debug(+myList);

//List with sObjects

//all the shops info from type utensils
List<Shop__c> ShopData = [select Shop_Name__c ,Timing__c,Location__c  from Shop__c where Shop_type__c='Utensils'];
system.debug(+shopData);



//List names of shops from shops List where shop type is NULL
List<Shop__c> ShopList = [select Shop_Name__c ,Timing__c,Location__c  from Shop__c where Shop_type__c = null ];
```

```
system.debug('List of shops where shop type is not Specified:'+ShopList);



//SET

//define a set
Set<integer> setList = new Set<integer>();

//add an element to the set
setList.add(1);
setList.add(8);
setList.add(3);
setList.add(10);
setList.add(111);



//Assert (assure) whether the set contains th element entered or not
System.assert(setList.contains(111));

//Remove the element from the set
setList.remove(1);
system.debug('Set is as:'+setList);

//Map

//define a map
Map<Integer,String> m = new Map<Integer,String>();
```

```
//insert values
//Map<String,String> currencies = new Map<String,String>('USA' => 'Dollars','INDIA'=> 'Rupees');

//Insert a key-value pair in map
m.put(1,'First Entry');
m.put(2,'Second Entry');
m.put(3,'Third Entry');
m.put(4,'Fourth Entry');
m.put(5,'Fifth Entry');
system.debug('Current status of map:'+m);

//Assert that map contains a key
 System.assert(m.containskey(2));

//Retrieve a value given a particuar key
String value= m.get(1);
system.debug('The value retrieved is:'+value);

//Return a set tht contains all of the keys in the map
Set<integer> KeysSet= m.keySet();
system.debug('Set of Keys:'+KeysSet);



Map<String,String> m = new Map<String,String>();
m.put('USA','Dollar');
m.put('India','Rupees');
system.debug('Map:'+m);

//sObject
```

```
//code to create an entry in object and then check whether specific entry is correct or not
Shop__c s = new Shop__c(Shop_Name__c = 'Keerti Stationary Store',Location__c='Old Panvel',Timing__c='8am to 11pm',Shop_type__c='Stationary');
insert s;
String chkVal=s.Shop_Name__c;
system.debug('The entered shop name is:'+chkVal);

if(chkVal == 'Keerti Stationary Store')
   system.debug('Correct Shop Entered');
else
   system.debug('Shop name is incorrect');


//code to check whether Location=null , if so delete the record
Shop__c ShopRec = [select Location__c from Shop__c where Shop_Name__c='Anand Sweets & Cakes'];
String ShopLoc = ShopRec.Location__c;
if(ShopLoc == null)
   delete ShopRec;
else
   system.debug('The shops Name is:'+ShopRec);


//-------------------------------------------------------------------------------------------------------------------------
//Variable Declaration: Using variables to store 3 numbers and adding them
Integer num1=(10*11)+30;//140
Integer num2=5000/5;//1000
Integer num3=800;
Integer res = num1+num2+num3;
system.debug('First Number is:'+num1);
```

```
system.debug('Second Number is:'+num2);
system.debug('Third Number is:'+num3);
system.debug('Addition of All Numbers is:'+res);
//or
system.debug('Addition Result:'+(num1+num2+num3));

//variable declaration: using vatiables to calculate simple interest
//S.I = (P*R*Y)/100
integer principle=1000;
integer rate=10;
integer year=5;
//double SimpleInterest=(principle*rate*year)/100;
decimal SI = (principle*rate*year)/100;
//system.debug('Simple Interest :'+SimpleInterest);
system.debug('Simple Interest ='+SI);

//date data type
Date newDate=Date.newInstance(2019, 9, 21);
system.Debug('Date is-'+newDate);

//display today's date
Date todayDate = Date.today();
system.debug('Today\'s Date is :'+todayDate);

//Add 2 years to today's date
Date todayDate = Date.today();
system.debug('Today\'s Date is :'+todayDate);
Date newDate=todayDate.addYears(2);
system.debug('Date after two years :'+newDate);
```

```
//add 4 months to today's date
Date todayDate = Date.today();
system.debug('Today\'s Date is :'+todayDate);
Date addMonths = todayDate.addMonths(4);
system.debug('Date after 4 months:'+addMonths);

//Calculate days between two dates
Date date1=Date.newInstance(2019,7,3);
Date date2=Date.newInstance(2019,7,23);
Integer NumofDays= date1.daysBetween(date2);
system.debug('Days Between these two dates are:'+NumofDays);

//Add 4 days to todays date
Date todayDate = Date.today();
system.debug('Today\'s Date is :'+todayDate);
Date AddDay = todayDate.addDays(4);
system.debug('Date after adding days :'+AddDay);

//check month and year of dtae
Date d1 = date.newInstance(2019,09,21);
system.debug('Today Date:'+d1);
system.debug('Month is:'+d1.month());
system.debug('Year is:'+d1.year());

//check whether year is Leap Year or not
Date d1 = Date.newInstance(2019, 04, 09);
Boolean LeapYr = Date.isLeapYear(d1.year());
system.debug('Is 2019 a Leap Year : '+LeapYr);
```

```
//Declare time in apex
Time mytime = Time.newInstance(5, 30, 15, 40);
system.debug('Time is:'+mytime);

//Add hour and minute to the time
Time mytime = Time.newInstance(6,12,4,40);
Time addHr = mytime.addHours(4);
time addMin = addHr.addMinutes(50);
system.debug('New Time: '+addMin);

//Datetime data type
Datetime d1 = Datetime.newInstance(2019,9,21,1,47,46);
String Date_Time=d1.format();
system.debug('Current Date-Time is:'+Date_Time);
system.debug('date1:'+system.now());//time from GMT
system.debug('date1:'+system.now().format());//org time
system.debug('date1:'+d1);

///add 2 hrs
Datetime d2 = d1.addHours(2);
system.debug('date2:'+d2);

//add 4 days
Datetime d3=d2.addDays(4);
system.debug('Date3:'+d3);

//add months
Datetime d4 = d3.addMonths(5);
```

```
system.debug('Date 5 :'+d4);

//add years
Datetime d5 = d4.addYears(2);
system.debug('Date 6:'+d5);

//fetch only date part from datetime value
Datetime datetym = datetime.newInstance(2019,6,12,10,2,00);
Date dt = datetym.date();
system.debug('Date is:'+dt);

//-----------------------------------------------------------------------------------------------------------------------

//STRING
String st1 = 'Sonal';
String st2 = 'Kumar';
Integer index = st2.indexOf('z');
system.debug('Index of letter \'m\' is:'+index); //4

String s1 = 'SONAL KUMAR';
String s2 = 'KUMAR';
system.debug('Index of \'KUMAR\' is:'+s1.indexOf(s2));//6

//add two strings by using '+'
String s1 = 'Komal';
String s2 = 'Sonal';
system.debug('Concatenated String is:'+(s1+s2));

String num1 = '22';
```

```
String num2 = '55';
system.debug(num1+num2);

//equals() method checks whether both strings are same or not
String str1 = 'Sonal';
string str2 = 'sonal';
boolean value = str1.equal(str2);
system.debug('Are both strings same?:'+value);



//CompareTo() method compares two strings lexicographically (dictionary order)
String s1 = 'abcd';
String s2 = 'efgh';
String s3 = 'abcd';
system.debug(s1.compareTo(s3)); //0

String s1 = 'abcd';
String s2 = 'efgh';
system.debug(s1.compareTo(s2));//-ve o/p -4

String s1 = 'abcd';
String s2 = 'efgh';
system.debug(s2.compareTo(s1));//4

String s1 = 'ad';
String s2 = 'bc';
system.debug(s2.compareTo(s1));

String s1 = 'deepika khanna';
```

```
Integer Valindex = s1.indexOf('nn');
system.debug('Index of \' nn \' is:'+Valindex);//11
Integer Valindex1 = s1.indexOf('a');
system.debug('Index of \'a \' is: ' +Valindex1);//6

//capitalize only the first letter of the string using capitalize() method
String s1 = 'sonal kumar';
String s2 = s1.capitalize();
system.debug(s2);//Sonal Kumar

String s1 = 'sonal Kumar';
String s2 = s1.capitalize();
system.debug(s2);//Sonal Kumar

//equals , ignore its case sensitivity
String s1 = 'SONAL';
String s2 = 'sonal';
boolean res1 = s1.equals(s2);
system.debug('Is S1 & s2 Same ? :'+res1);//false
//ignore case
boolean res2 = s1.equalsIgnoreCase(s2);
system.debug(res2);//true

//to upper case an lower case
String s1 = 'kumar';
String res = s1.toUpperCase();
system.debug('Capital Form:'+res);//KUMAR

//lowercase
```

```
String s1 = 'KUMAR';
String res1 = s1.toLowerCase();
system.debug('Capital Form:'+res1);

//concatenate two string
String nm1 = 'Sonal';
String nm2 = 'Kumar';
String name = nm1+nm2;
system.debug('Name is:'+name);

//contains method
String name = 'Sonal';
system.debug('Using contains method : '+name.contains('son'));

//Also
String nm = 'Sonal';
String nm1 = 'Hello';
String nm3 = 'Son';
Boolean res = nm1.contains(nm);
system.debug(res);//false
Boolean res2 = nm.contains(nm3);
system.debug(res2);//true

//=== operator
//if x,y references to same exact location then o/p is true
String x = 'Sonal';
String y;
x=y;
system.debug(x===y);
```

```
//Operators in Apex
Integer result = 100;
result +=10;//110
system.debug('Addition:'+result);
result -=10;//100
system.debug('Subtraction:'+result);
result *=10;//1000
system.debug('Multiplication:'+result);
result /=5;//200
system.Debug('Division: '+result);

//------------------------------------------------------------------------------------------------------------------------
//Trailhead Execution

//create list and add elements in one step
List<String> colors = new List<String> { 'red', 'green', 'blue' };
system.debug(colors.get(1));

//creating array,then converting it to list
Integer[] num = new List<Integer>{1,2,3};
system.debug(num[0]);

//system.assertEquals use
//
List<String> colors = new List<String> { 'red', 'green', 'blue','red'};
String color1 = colors[0];
String color2 = colors[3];
system.assertEquals(color1, color2);
```

```
//list can also be created as
List<String> lst = new String[10];
//we can also add elementsin list as
lst[0]='ABC';
lst[2]='FGH';
//loop list
for(Integer i=0;i<lst.size();i++)
{
   system.debug('List items are:'+lst[i]);
}
system.debug(lst.size());

//remove element frm list
lst.remove(0);
for(Integer i=0;i<lst.size();i++)
{
   system.debug('List items are:'+lst[i]);
}

//set() method and clone() method
List<String> lst = new List<String>{'Sonal','Komal','Nilu'};
List<String> lst1 = lst.clone();
lst1.set(1,'Sammy');
system.debug('Original List:'+lst);
system.Debug('Duplicate List:'+lst1);

//sort list in ascending order
List<String> lst1 = new List<String>{'Sonal','Komal','Nilu'};
```

```
lst1.sort();
system.debug('Sorted list is:'+lst1);//(komal, Nilu,Sonal)

//check whether list is empty or not answer will be in true or false
List<String> lst = new List<String>{'Sonal','Komal','Nilu'};
Boolean emptylst=lst.isEmpty();
system.debug('Is List Empty:'+emptylst);//false

//clear all elements in the list
List<String> lst = new List<String>{'Sonal','Komal','Nilu'};
lst.clear();
system.debug(lst);

Boolean emptylst=lst.isEmpty();
system.debug('Is List Empty:'+emptylst);//true



//------------------------------------------------------------------------------------------------------------------------
List<Integer> matrix = new List<Integer>();
matrix.add(new List<Integer>{1,2,3});
matrix.add(new List<Integer>{4,5,6});
matrix.add(new List<Integer>{7,8,9});

//this is how you iterate through two dimensional array.
for(Integer i=0;i < matrix.size();i++)
   for(Integer j=0;j < matrix[i].size();j++)
       {
          Integer val = matrix[i][j];
          System.debug(val);
```

```
        }

//-------------------------------------------------------------------------------------------------------------------

List<List<Integer>> listOdListInt = new List<List<Integer>> { {0, 1, 2, 3}, {3, 2, 1, 0}, {3, 5, 6, 1}, {3, 8, 3, 4}
};

Integer i = listOdListInt.get(1).get(0); // which returns integer from 0th index of 1st index list

List<List<Integer>> matrix = new List<List<Integer>>();
List<Integer> l1 = new List<Integer>{1,2,3};
List<Integer> l2 = new List<Integer>{4,5,6};
List<Integer> l3 = new List<Integer>{7,8,9};
matrix.add(l1);
matrix.add(l2);
matrix.add(l3);
system.debug('Original list is: '+ matrix);

for(Integer i=0;i < matrix.size();i++)
   for(Integer j=0;j < matrix[i].size();j++)
        {
           Integer val = matrix[i][j];
           System.debug(val);
        }

//-------------------------------------------------------------------------------------------------------------------
//write a method in apex which will have a 2D list of integers ,this list will be tranposed.
public class twodlistandtranspose {
   public  void main() {
      List<List<Integer>> twodlist = new List<List<Integer>>();
```

```
        List<Integer> l1 = new List<Integer>{1,2,3};
        List<Integer> l2 = new List<Integer>{4,5,6};
        List<Integer> l3 = new List<Integer>{7,8,9};
        twodlist.add(l1);
        twodlist.add(l2);
        twodlist.add(l3);
        system.debug('Original list is: '+ twodlist);
        for(integer i=0; i<3; i++) {
           for(integer j=0; j<3; j++) {
              if(i < j) {
                 integer temp = twodlist[i][j];
                 twodlist[i][j] = twodlist[j][i];
                 twodlist[j][i] = temp;
              }
           }
        }
        system.debug('List after transpose is: '+ twodlist);
    }
}

//------------------------------------------------------------------------------------------------------------------------
//Sets
//initialize set
Set<Integer> s1 = new Set<Integer>();

//add values to set. Values will be added in random order and no duplicate values will be added
s1.add(2);
s1.add(4);
s1.add(8);
```

```
s1.add(4);

//display set
system.debug('Set is:'+s1);

//All the above code can be optimized as
Set<Integer> s2 = new Set<Integer>{2,4,6,8,2};
system.debug('Set is:'+s2);

//------------------------------------------------------------------------------------------------------------------------
//if-else

integer score = 90;
if(score<10)
   system.debug('Work Hard!!');
else
   system.debug('Congrats!!');

//check rank of student based on scores

integer score=99;
if(score>70 && score<=80)
{
   system.debug('3rd rank');
}
else if(score>80 && score<=90)
{
   system.debug('2nd rank');
}
```

```
if(score>90 && score<=100)
{
   system.debug('1st rank');
}
else
{
   system.debug('Not in top 3 Ranks');
}

//check whether no. is -ve, +ve or zero
integer num=0;
if(num>0)
{
   system.debug('No. is positive');
}
else if(num<0)
{
   system.debug('No. is negative');
}
else
{
   system.debug('Number is Zero');
}

//medal scored in race

Integer position=4;
String medal_color;
```

```
if(position == 1)
{
   medal_color='Gold';
}
else if(position == 2)
{
   medal_color='Silver';
}
else if(position == 3)
{
   medal_color='Bronze';
}
else
{

}
if(medal_color != null)
{
   system.debug('You have scored a '+ medal_color + ' medal');
}
else
{
   system.debug('You have scored no medal');
}

//while loop
//print your name 10 times
integer i = 1;
while(i<=10)
```

```
{
   system.debug('Sonal Kumar');
   i++;
}

//print numbers 1 to 10
integer num=1;
while(num<=10)
{
   system.debug(num);
   num++;
}

//same using for loop, print numbers 1 to 10
for(integer i=1;i<11;i++)
{
   system.debug(i);
}

//display sequence 10,8,6,4,2,0
for(integer i = 10; i >=0; i-=2)
{
   system.debug(i);
}

//a List contains names of all employees display it using for loop
List<String> empname = new List<String>{'Komal','Sonal','Suman','Sona'};
for(String names : empname)
{
```

```
    system.debug('Names are: ' + name);
}

//break and continue
for(integer i = 1; i<=10; i++)
{
    if(i == 5)
    {
        break;
    }
   system.debug(i);
}

for(integer i = 1; i<=10; i++)
{
    if(i == 5)
    {
        continue;
    }
   system.debug(i);
}

//
if('Hello'.endsWith('o'))
{
    system.debug('me');
    system.debug('me too!');
}
```

```
//-------------------------------------------------------------------------------------------------------------------------
//nested loop
for(integer i=0;i<=3;i++)
{
   for(integer j=0;j<=2;j++)
   {
      system.debug('i= ' +i +' b=' +j);
   }
}

//pattern printing
for(integer i=1; i<=4; i++)
{
   for(integer j=1; j<=i ; j++)
   {
      system.debug(i);
   }
   //system.debug('\n');
}

//pattern2
for(integer i=1; i<=4; i++)
{
   for(integer j=1; j<=i ; j++)
   {
      system.debug(j);
   }
}
```

```
//pattern 3
for(integer i=1;i<=16;i+=3)
{
   system.debug(i);
}

//pattern 4
for(integer i=20;i>=5;i-=5)
{
   system.debug(i);
}

//pattern 5
for(integer i=2;i<=10;i+=2)
{
   system.debug(i);
}

//------------------------------------------------------------------------------------------------------------------
//creating actual dog object to access Dog class variables and methods

Dog d1 = new Dog();
Dog d2= new Dog();
d1.name = 'Scooby';
d1.age=12;
d1.display();
d2.name='Boxer';
d2.age=3;
d2.display();
```

```
//-----------------------------------------------------------------------------------------------------------------
//creating actual Employee object to access employee class variables and methods
Employee e1 = new Employee();
Employee e2 = new Employee();
Employee e3 = new Employee();
Employee e4 = new Employee();
e1.name = 'Shaurya';
e1.desig = 'Manager';
e1.disp();
e2.name = 'Saksham';
e2.desig = 'Team Lead';
e2.disp();
e3.name = 'Sheetal';
e3.desig = 'Intern';
e3.disp();
//-----------------------------------------------------------------------------------------------------------------
//Testing static and non static methods
//static methods can be called as
StaticExp.method1();
StaticExp.method2(); //non static method
StaticExp s=new StaticExp();
s.method2();

//static variable
CatClass c1= new CatClass();
CatClass c2 = new CatClass();
c1.name='Sweety';
c2.name='Rajjo';
```

```
CatClass.count=5;
c1.disp();
c2.disp();
CatClass.count++;
system.debug('Count:'+CatClass.count);

//-----------------------------------------------------------------------------------------------------------
//Access Modifier
Cats c = new Cats();
c.setSize(-1);
c.setName('Pluto');
c.disp();

//=----------------------------------------------------------------------------------------------------------
//constructor example
ConstructorExp c = new ConstructorExp(10,'Rosy');
c.disp();
ConstructorExp c2 = new ConstructorExp();
c2.disp();

//----------------------------------------------------------------------------------------------------------
//Inheritance Example
InheritanceExp e = new InheritanceExp();
Truck e1= new Truck();
e1.model();
e1.speed();
e.speed();

//----------------------------------------------------------------------------------------------------------
```

```
//Collections in Apex
//List

//create a List
List<String> lstnames = new List<String>();
//add names to list
lstnames.add('Sonal');
lstnames.add('Ajit');
lstnames.add('Komal');

//display the list
system.debug('Names of friends are: '+lstnames);

//add a name at a specific position
lstnames.add(0,'Arun');
system.debug('Names of friends are: '+lstnames);

lstnames.add(1,'Justin');
system.debug('Names of friends are: '+lstnames);

//remove justin's name which is at index 1
lstnames.remove(1);
system.debug('Names of friends are: '+lstnames);

lstnames.add('Sonal');
system.debug('Names of friends are: '+lstnames);

//another way to create list
String[] lst = new List<String>{'one','two','three'};
```

```
    system.debug(lst);


//display a particular array element
system.debug('Element at index 2:'+lst[2]);
//or
system.debug('Element at index 1:'+lst.get(1));


system.debug('Size of list:'+lst.size());
```