APEX KODLAR DERS NOTLARI

APEX BASIC

```
String z = 'lovelace';
System.debug(z.length());
String y = 'candyce';
System.debug(z + ' ' + y);
System.debug(y.reverse().toUpperCase());
System.debug(z.capitalize());

Integer x;
x = 58;
System.debug(x);
Integer w = 98;
System.debug(x + w * (x + w));

Double v = 298;
System.debug(v);

Decimal u = 10.27;
System.debug(u);
Decimal ua;
Decimal t;
Decimal h;
t = 5;
h = 7;
ua = (t * h) / 2;
System.debug(ua);

Date s = Date.newInstance(2022, 06, 27);
System.debug(s);
System.debug(s.format());
Integer r = Date.Today().year();
System.debug(r);
Integer q = 1974;
Integer p = r - q;
System.debug(p);
String o = String.valueOf(p);
System.debug(o);
```

List , Set , Map

```
List <String> myList = new List <String> ();
myList.add('ahmet');
myList.add('mehmet');
myList.add('ayse');
myList.remove(1); // index numarasına göre listeden elemanı kaldırıyor (mehmet)
system.debug(myList);

List <String> myList1 = new List <String>{'ela', 'seda','zeynep'}; // listeye elaman ekleme 2.yöntem
myList1.set(0,'veda');  // 'set'=ayarla,  belirli bir indexe istediğin bir elemanı atayabilirsin
system.debug(myList1); // 1 .index i getir
 system.debug(myList1.get(1)); // 'get' index i 1 olan veriyi getir

String [] myList2= new String[4];   // 4 elemanlı bir liste oluşturduk
myList2[0]='adem';   // listenin 2.elemanı olarak adem i ekladik
myList2[2]='bulent';  //  listenin 4.elemanı olarak Bülent hocayı ekledik
myList2.add('dsa'); // 'add' listenin sonuna ekleme yapar
system.debug(myList2);

List <Integer> myNum = new Integer [5];
myNum.add(0,25);
myNum.add(1,27); // listenin sayısını artırıyor
myNum.set(4,50); // listenin sayısı aynı kalıyor, belirlenen indexin değerini değiştiriyor.
system.debug(myNum);

List<String> listem = new List<String>{'ali','b'};
Boolean result = listem.contains('ali');   // 'contains' listenin içinde varmı, true-false döndürür
system.debug(result);
List<String> listem = new List<String>{'ali','b'};
```

```
system.debug(listem.indexof('b'));   // 'indexof' listenin elemanın index numarasını getirir

Set<String> mySet = new Set<String>();  // listeyi başta set olarak (unique) olarak tanımladık.
mySet.add('kamil');  // duplicate yapılan kayıtlar listeye eklenmiyor.
mySet.add('ahmet');
system.debug(mySet);
system.debug(mySet.size());  // 'size' listenin boyutunu gösterir
system.debug(mySet.isempty());  // 'isempty' liste boş mu, true-false dönüyor
String abc ='nasılsınız';
system.debug(abc.length());

Map<Integer, String> myMay= new Map <Integer, String> ();      // key=value
myMap.put(1,'bulent hoca');
myMap.put(2,'aysen');
system.debug(mymap);
system.debug(myMap.get(1));

Map<Integer, String> myMap = new Map<Integer, String>();
myMap.put(1,'bulent'); //
myMap.put(2,'aysen');
system.debug(mymap);
system.debug(mymap.remove(1));
system.debug(mymap.get(0));
system.debug(mymap);

Map<String, String> myMap = new Map<String, String>{'a'=>'veli', 'b'=>'deli'};   // => key ve value eşitlemesini sağlıyor.
system.debug(mymap);
```

Arithmetic Operators,

```
/Arithmetic Operators
//(+, - , *, /)

/*Decimal x = 5;
Decimal y = 8;

Decimal z = y / x;

system.debug(x + y);
system.debug(x - y);
system.debug(x * y);
system.debug(z);*/

//Assignment Operators

Integer x = 5;
x += 3;    //x = x +3;
system.debug(x);

Integer y = 10;
y -= 5;    //y = y - 5;
system.debug(y);

Integer z = 10;
z *= 5;    //z = z * 5;
system.debug(z);


Integer d = 10;
d /= 5;    //d = d / 2;
system.debug(d);
```

Assignment Operators,

```
```

Comparison Operators,

```
//Comparison Operators

Integer x = 5;
Integer y = 5;
```

```
String d = 'deli1';

String e = 'deli';

System.debug(d <> e);

System.debug(x==y);       //==

System.debug(x>y);   //     > , <

System.debug(x>=y);   //    >= , <=

System.debug(x != y);   //  !=

System.debug(x <> y);

System.debug(!True);

System.debug(!False);
```

Logical Operators,

```
//Logical Operators

// AND => &&  -----   OR => ||

System.debug(True && True);   // True
System.debug(True && False);   // False
System.debug(False && True);   // False
System.debug(False && False);  // False

System.debug(True || True);      // True
System.debug(True || False);      // True
System.debug(False || True);      // True
System.debug(False || False);      // False

// Logical Operators AND = && --/-- OR = ||

// AND / && ---------------------------------------

System.debug(' 2 tane True olursa :  ' + (True && True));
System.debug(' 2 tane True olursa :  ' + (2>1 && 2==2));

System.debug(' 1 tane True 1 tane False olursa :  ' + (True &&False));
System.debug(' 2 tane False olursa :  ' + (False &&False));

// OR / || ---------------------------------------

System.debug(' 1 tane True 1 tane False olursa :  ' + (True &&False));
System.debug(' 2 tane False olursa :  ' + (False &&False));
System.debug(' 2 tane True olursa :  ' + (True &&True));
```

Math methods..

```
// Math Operator

integer x= 121;
integer y=111;
integer m=-64;

integer z = Math.max(x, y); // max
System.debug('Max sayı : ' + z);
integer a = Math.min( x, y); // min
System.debug('Min Sayı : ' +a);
Decimal b = Math.sqrt( x); // Karekök
System.debug('x in Karekökü : ' +b);
integer c = Math.abs(m); // Mutlak değer
System.debug('m in mutlak değeri : ' +c);
integer d = Math.mod(13,5); // mod
System.debug('13 ün 5 e bölümünden kalanı verir : ' +d);
Double e = Math.pow(5,2); // üstü değeri
System.debug('5 üstü 2 yi hesaplar : ' +e);
System.debug(Math.pow(5,2));
```

```
Integer a = 10;
a = 7;
System.debug(a);
Enum seasons{winter, spring, summer, fall}  // when once set the enum, the content doen't change.
System.debug(seasons.winter);
System.debug(seasons.values());
Final Decimal pi = 3.14;
// pi = 4.13;
System.debug(pi);  // causes error, because a finalized variable cannot be changed.
Integer b = 10;  // This is an assignment, not an equality.
String c = 'HKubra';  // a string assignment.
Integer d = 20;
System.debug(b == d);  // returns FALSE
System.debug(b != d);  // returns TRUE
System.debug(b < d);  // returns TRUE
System.debug(b > d);  // returns FALSE
System.debug(b <> d);  // returns TRUE
Integer e = 30;
System.debug((b > d)  && (e < b));  // returns FALSE
// if there is any FALSE value in the text, than AND operator returns FALSE
// with AND operator TRUE result is possible only if all statements are TRUE
System.debug((d > b) || (e < b));  // returns TRUE
// if there is any TRUE value in the text, than OR operator returns TRUE
// with OR operator FALSE result is possible only if all statements are FALSE
System.debug(b + d);  // returns 30
System.debug(b - d);  // returns -10
System.debug(b * d);  // returns 300
System.debug(b / d);  // returns 0, because we defined integers and it rolls to the whole number stated before the decimal point
Decimal f = 10;
Decimal g = 4;
System.debug(f / g);  // returns 2.5 as expected
Integer h = 5;
Integer i = 8;
h += 5;
System.debug(h);  // returns 10
i *= 2;
System.debug(i);  // returns 16
i++;
System.debug(i);  // returns 17, because it was multiplied by 2 in previous step
Integer j = 5;
j++;
System.debug(j);  // adds 1 to the variable and returns 6
++j;
System.debug(j);  //  again added 1 to the variable in line 43 and returns 7
System.debug(j++);  // also returns 7, because, firstly system.debug runs, after running system.debug it adds and we don't see this additio
System.debug(++j);  // returns 9, beacause it was added 1 in previous step, and again added 1 here
Integer k = 64;
Integer l = -25;
System.debug(Math.max(k, l));  // returns the highest value
System.debug(Math.min(k, l));  // returns the lowest value
System.debug(Math.sqrt(k));  // returns 8
System.debug(Math.abs(l));  // returns the absolute value, 25; not -25.
System.debug(Math.mod(k, l));  // returns the modulus which is the remainder of the division operation
System.debug(Math.pow(l, 2));  // returns the second power of variable l, that is, l is squared.
Integer m = 5;
Decimal newM = m;
System.debug(newM);  // converted from integer to decimal type as another variable
Decimal n = 5.2;
Integer newN = (integer)n;
System.debug(newN);  // converted from decimal to integer type but data lose occurs here
String o = '10';
String p = '55';
System.debug(o + p);  // concatenates the strings and returns 1055 string value
Integer convertedO = Integer.valueOf(o);  // converts string to integer with a new variable assignment
Integer convertedP = Integer.valueOf(p);
System.debug(convertedO + convertedP);
Integer q = 96;
String r = String.valueOf(q);  // converts integer to string with a new variable assignment
System.debug(r + 1);  /* concatenates the string to integer 1,
and returns 961.  But this is not Nine  Hundred Sixty One, but Ninety Six One! */
Date s = date.newInstance(2022, 06, 29);  // injects the specified date with time
System.debug(s);
String t = String.valueOf(s);  // converts date to string
System.debug(t + '3');
```

if -else

```
// a<b, a<=b , a>b ; a==b , a!=b,
Integer a = 5;
Integer b = 10;
if (a < b ){

   System.debug('a < b ');

}else if(False){

    System.debug('If In 2 ');

}else{
     System.debug('If In 3 else ');

}

System.debug('If Out');


// a<b, a<=b , a>b ; a==b , a!=b,
Integer a = 5;
Integer b = 10;

String result = (a>b) ? 'True a < b' : 'False a < b ' ;


System.debug(result);
```

Switch

```
// Switch

Integer day = Math.mod(15,7);// result = 1

Switch on day{
    when 1 {
        System.debug('pazartesi');
    }
    when 2{
        System.debug('sali');
    }
    when 3{
        System.debug('carsamba');
    }
    when 4{
        System.debug('persembe');
    }
    when 5{
        System.debug('cuma');
    }
    when 6{
        System.debug('cumartesi');
    }
    when else {
        System.debug('pazar');
    }


}
```

FOR

```
Integer i;


for ( i = 5 ; i < 100; i*=5){
    System.debug('i dongusu' + i);


}
```

While - do while

```
/*Integer i = 0 ;

while (i<10){
    i++;
    System.debug('i  while dongusu : '+i );

    i+=5;

}*/

Integer i = 0 ;

while (i<10){

    System.debug('i  while dongusu : '+i );

    i+=5;

}

//do - while
Integer i = 2;


do {

    system.debug('1 key calistir'+ i);
    i++;
}while(i<5);
```

break - continue

```
/*for(integer i = 0 ; i<5 ; i++){

    if(i == 3){

        break;   //burada sistem, dongu kapaniyor

    }
    System.debug('i : '+ i);
}*/

for(integer i = 0 ; i<5 ; i+=3){

    if(i == 3){

        continue;   //burayi atlayip dongu devam ediyor

    }
    System.debug('i : '+ i);
}
```

APEX TEST

4 ISLEM

APEX CLASS

```
public class mavi {
    public Integer sum(Integer x, Integer y){
        Integer result =x+y;
        return result;
    }
public Integer sub(Integer x, Integer y){

        Integer result =x-y;
        return result;
}
    public Integer mul(Integer x, Integer y){

        Integer result =x*y;
        return result;
    }
    public Decimal div(Decimal x, Decimal y){

        Decimal result =x/y;
        return result;
    }
}
```

APEX TEST

```
@isTest
public class mavi_Test {
    @isTest
    public static void  sum_Test(){

        mavi testhesap=New mavi();
        Integer testresult = testhesap.sum(5,7);
        System.assertEquals(12,testresult,'not true');
    }
@isTest
    public static void  sub_Test(){

        mavi testhesap=New mavi();
        Integer testresult = testhesap.sub(10,7);
        System.assertEquals(3,testresult,'not true');
    }
@isTest
    public static void  mul_Test(){

        mavi testhesap=New mavi();
        Integer testresult = testhesap.mul(5,3);
        System.assertEquals(15,testresult,'not true');
    }
   @isTest
    public static void  div_Test(){

        mavi testhesap=New mavi();
        Decimal testresult = testhesap.div(10,5);
        System.assertEquals(2,testresult,'not true');
    }
}
```

```
DML PRACTICE
```

DML PRACTICE

```
List<List<sobject>> results=[find 'united' returning Account(id,name),contact];
system.debug(results); // Sosl klasik veritabanından veri getirme sorgusu

// Search.query  method kullanarak aynı işlemin yapılması
String var1 = 'find {united} returning Account(id,name),contact';
List<List<sobject>> results=search.query(var1);// method string parametresi kabul eder
system.debug(results);
```

```
// insert
 Contact con1 = new Contact(lastname='contact1');
contact con2 = new Contact();
con2.lastname='contact2';
insert con1;
insert con2;
// update
contact con1update= [select lastname from contact where name ='contact1'];
con1update.lastname='yenicontact1';

contact con2update= [select lastname from contact where name ='contact2'];
con2update.lastname='yenicontact2';

list<contact> updatedcons = new List<Contact>();
updatedcons.add(con1update);
updatedcons.add(con2update);

update updatedcons;

// upsert
List<contact> upsertList = new List<contact>();

contact con3 = new contact(lastname='contact3');
upsertList.add(con3);

contact con1guncelle =[select lastname from contact where name='yenicontact1'];
con1guncelle.lastname='guncelcon1';

upsertList.add(con1guncelle);

upsert upsertList;

//merge
contact con3=[select lastname from contact where name='contact3'];
contact con2=[select lastname from contact where name='yenicontact2'];

merge con2 con3;
// delete
contact con2=[select lastname from contact where name='yenicontact2'];
system.debug(con2);
delete con2;

// undelete
contact con2=[select lastname from contact where name='yenicontact2' all rows];
undelete con2;

//Database.insert();/upsert/merge/delete/undelete
contact con1= new contact(lastname='yeni');
contact con2= new contact();
List<contact> newList = new List<contact>();
newList.add(con1);
newList.add(con2);
database.insert(newList,false);

SQL Dynamic Query Example:
String var2 = 'united';
List<Account> accs = [select id,name from account where name like:'%'+ var2 +'%'];
system.debug(accs);


String var2 = 'GenePoint';
List<sobject> accs= database.query('select id,name from account where name  =:var2');
system.debug(accs);
```

SOQL PRACTICE

```
//select fields(All) from contact where lastname ='Song' limit 200
//select name from account where name like '%oil%'
find {new}/ find {new*}

find {new} limit 10
find {new} limit 13
find {new*}  in all fields returning account,contact
find {new*}  in all fields returning account(name),contact(lastname)
find {c?a*}
find {c?a*}  returning account(name,phone,description)
find {r?s*} returning account (name)
find {oil} in name fields returning account(name),opportunity(name)
find {oil} in name fields returning account(name limit 1),opportunity(name where name like '%in%')
find {oil} in name fields returning account(name order by name desc),opportunity(name where name like '%in%')
find {al*} returning expense__c(name)

Apex/DMl examples
find {new*} returning contact(name,description),Account// developer
 in Apex :
List<List<sobject>> result=[find 'new*' returning contact(name,description),Account];
system.debug(result);
List<contact> returnedcontacts=result[0];
system.debug(returnedcontacts.get(0).name); Listenin ilk elemanının ismini döndürür.
List<contact> updList = new List<contact>();
for(contact con :returnedcontacts){
    con.description='abc';
    updList.add(con);
    system.debug(con.description);
}
system.debug(updList.size());
update updList;
system.debug(updList);

contact x=returnedcontacts.get(0);// ekstra contact manipule etme
x.lastname='King Artur';
system.debug(x);
```

APEX ODEV

APEX TRIGGER

```
trigger AccBeforeInsertUpdate on Account (before insert, before update) {

    for(Account acc:Trigger.new){

        if(Trigger.isinsert){
            acc.Description='bu fild before insert ile olusturuldu';
        }
        if(Trigger.isupdate){
            acc.Description='bu fild before update ile olusturuldu';
        }
    }
}
```

0pprtunityde yapilan bir degisiklikle Accountta bir degisiklik olusmasi

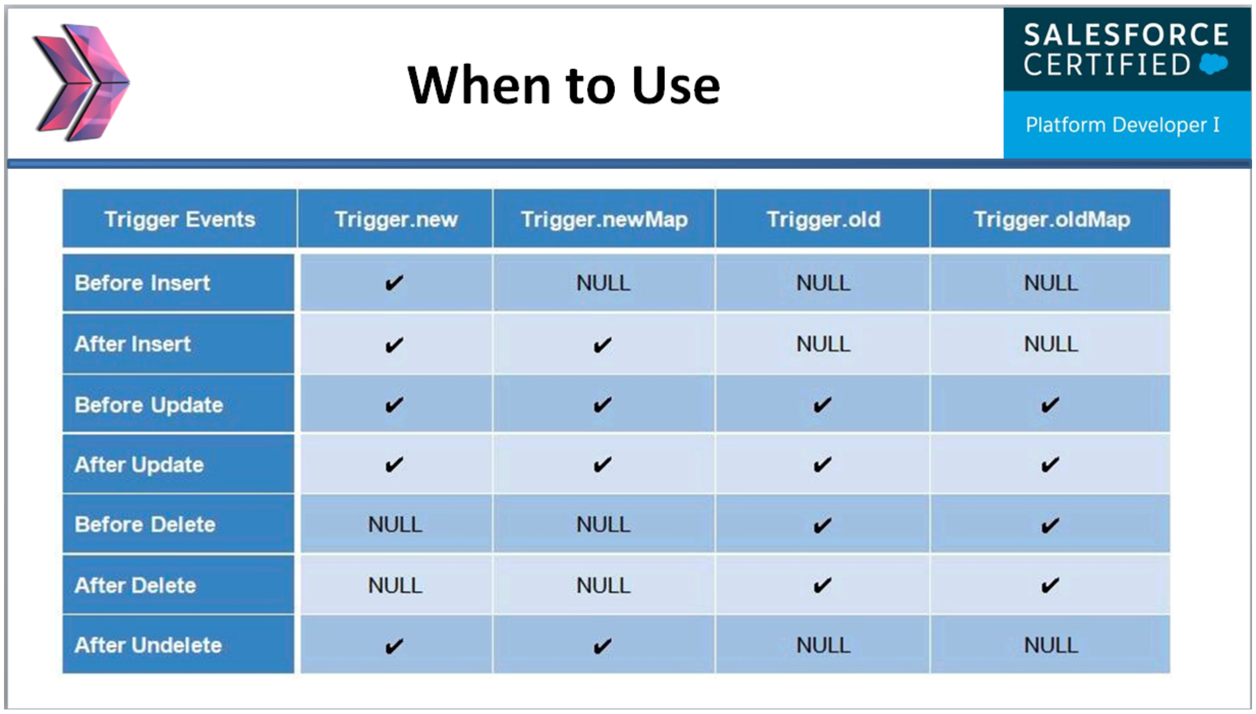
```
trigger AccAnd0pp on Account (before insert) {
   List<Opportunity>newList= new List<Opportunity>();
    for(Account acc:Trigger.new){
        if(acc.Industry=='Banking' && acc.Rating=='Hot'){
            opportunity opp1=new opportunity();
            opp1.Name=acc.name;
            opp1.StageName='closedwon';
            opp1.CloseDate=system.today()+5;
            newList.add(opp1);
        }
    }
     insert newList;

}
```

APEX TRIGGER PRACTICE

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/bb3ec0b6-f51d-44b3-88d1-bfc24fdf7d8a/Trigger_Practices1.docx

APEX TRIGGER-NEW-OLD-MAP


SALESFORCE CERTIFIED
Platform Developer I

When to Use

Trigger Events	Trigger.new	Trigger.newMap	Trigger.old	Trigger.oldMap
Before Insert	✔	NULL	NULL	NULL
After Insert	✔	✔	NULL	NULL
Before Update	✔	✔	✔	✔
After Update	✔	✔	✔	✔
Before Delete	NULL	NULL	✔	✔
After Delete	NULL	NULL	✔	✔
After Undelete	✔	✔	NULL	NULL

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/f181aeb5-04be-4761-8fbf-22829f69518b/Trigger_Practice_2.docx

APEX TRIGGER-HELPER CLASS

```
Trigger with Helper Class
//Trigger Code
trigger AccOppmultiEvent on Account (before insert,before update,after insert) {

    if(Trigger.isbefore && Trigger.isinsert){
        AccMultiTriggerClass.beforeInsert(Trigger.new);
    }
    else if(Trigger.isbefore && Trigger.isupdate){
        AccMultiTriggerClass.beforeUpdate(Trigger.new);
    }
    else if(Trigger.isafter){
        AccMultiTriggerClass.afterInsert(Trigger.new);
    }

}
//Helper Class Code
public class AccMultiTriggerClass {
    public static void beforeInsert(List<Account> accList){
        for(Account acc:accList){
            acc.Type='Prospect';
        }
    }
    public static void beforeUpdate(List<Account> accList1){
        for(Account acc1:accList1){
            if(acc1.Ownership=='Private'){
                acc1.Rating='hot';
            }

        }
    }
    public static void afterInsert(List<Account> accList2){
        List<Task> taskList = new List<Task>();
        for(Account acc2:accList2){

            if(acc2.AnnualRevenue >=100000000){

            Task t1= new Task();

            t1.Subject='Şirketle iletişime geç!';
            t1.Status='in progress';
            t1.Priority='normal';
            t1.WhatId=acc2.Id;
            taskList.add(t1);

            }



        }
        insert taskList;
    }
}
//Trigger Old Usage
trigger ContOld on Contact (before update) {

    for(contact con:Trigger.old){
        if(con.title ==null){
            for(contact con1:Trigger.new){
                   con1.title='CEO';
            }

        }
    }

}
//Trigger Old and New Map Usage

Trigger Example11 on Account (Before Update) {
Map<id,Account> oldmap = Trigger.oldMap;
Map<id,Account> newmap = Trigger.newMap;
for(id key : newmap.keySet()){
Account ol = oldmap.get(key);
Account nw = newMap.get(key);
if (ol.ownership == 'public' && nw.Ownership == 'private'){
nw.Phone='765536374646';
}
}
}
```