T.C.

MİLLÎ EĞİTİM BAKANLIĞI

BİLİŞİM TEKNOLOJİLERİ

TEMEL KOMUTLAR

Ankara, 2017

- Bu bireysel öğrenme materyali, mesleki ve teknik eğitim okul / kurumlarında uygulanan çerçeve öğretim programlarında yer alan kazanımların gerçekleştirilmesine yönelik öğrencilere rehberlik etmek amacıyla hazırlanmıştır.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR

ALAN	**Bilişim Teknolojileri**
DAL	**Veritabanı Programcılığı**
MODÜLÜN ADI	**Temel Komutlar**
MODÜLÜN SÜRESİ	40/80
MODÜLÜN AMACI	İş sağlığı ve güvenliğine dikkat ederek mobil işletim sistemi standartlarına göre temel komutları kullanabileceksiniz.
MODÜLÜN ÖĞRENME KAZANIMLARI	**1.** Veri tipi isimlendirme kurallarına ve ihtiyaca uygun veri tiplerini ve sabitleri kullanabileceksiniz. **2.** İşlem önceliğini dikkate alarak operatörleri kullanabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Uygun programların yüklü olduğu bir bilgisayar laboratuvarı. **Donanım:** Gerekli programların yüklü olduğu bir bilgisayar.
ÖLÇME VE DEĞERLENDİRME	Bireysel öğrenme materyali içinde yer alan ve her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendirebileceksiniz. Öğretmeniniz, bireysel öğrenme materyalinin sonunda, ölçme araçları (uygulamalı faaliyetler, iş ve performans testleri, çoktan seçmeli / doğru-yanlış ve boşluk doldurmalı sorular, vb.) kullanarak kazandığınız bilgi ve becerileri ölçüp değerlendirecektir

GİRİŞ

Sevgili Öğrencimiz,

Son yıllarda insanların gereksinimlerinde meydana gelen değişimlerle birlikte akıllı telefon kullanımı artmıştır. Bunun sonucunda mobil programlamanın dünyada ve ülkemizde oldukça talep gören bir alan olduğu tartışılmaz bir gerçek olmuştur. Hemen her konuda bir mobil program yazılarak akıllı cihazlarla hayatımız her konuda kolaylaştırılmaktadır.

Teknolojik hayatta yer edinilmek ve aranılan eleman olunmak isteniyorsa mobil programlama da hayatın bir parçası hâline getirilmek zorundadır.

Bu bireysel öğrenme materyaliyle tüm akıllı cihazlar için program yazma anlamında ilk adım atılmış olacaktır. Takip eden diğer bireysel öğrenme materyalleri de tamamlandığında akıllı cihazlar için mobil uygulamalar hazırlayabilir duruma gelinecektir.

ÖĞRENME FAALİYETİ–1

ÖĞRENME KAZANIMI

Veri tipi isimlendirme kurallarına, ihtiyaca uygun veri tipleri ve sabitleri kullanabileceksiniz.

ARAŞTIRMA

- Daha önceki yıllarda gördüğünüz programlama dillerinden de aşina olduğunuz veri tiplerinin neler olduğunu hatırlamak için bir araştırma yapınız.

- Birbirinden farklı programlama dillerinde farklı veri tipleri kullanılıyor mu? sorusuna bir cevap arayınız ve topladığınız bilgileri sınıfınızla paylaşınız.

1. TEMEL KOMUTLAR

Bilgisayarda program yazabilmek için birçok programlama dili kullanılmaktadır. Günümüzde akıllı cihazlarda kullanılmak üzere oluşturulan programları yazabilmek için genel olarak android studio, elipse ve java tabanlı derleyiciler gibi programlama unsurları kullanılmaktadır.

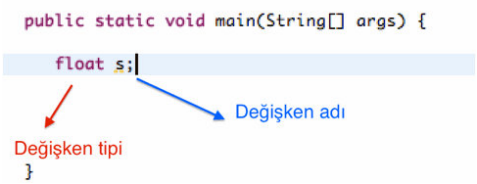
1.1.Veri Tipi İsimlendirme Kuralları

Değişkenler, atandıkları tipe bağlı olarak bellekte farklı biçimlerde tutulur ve bu tipler birbirlerinden farklı boyutlara sahiptir. Yazılımlar, genel olarak veriyi işleyip kullanıcıya sonucu sunacak şekilde tasarlanır. Bu süreçte kullanıcıdan da veri talep edilebilir ya da işlemler tamamen bellekte tutulan değerler üzerinden yapılabilir. Değişkenler yani veriler, RAM (Random Access Memory) üzerinde kısa süreli olarak saklanır.

Veri tipleri isimlendirildiklerinde bu adlar onların kimlikleridir. Daima o kimlikleriyle çağrılır. Başka bir deyişle adlar ilgili öğeye erişmeyi sağlayan işaretçilerdir. Verilen adlar, işletim sistemine ve derleyiciye bağlı olarak bazı kısıtlara tabidir.

Kullanılacak programlama diline göre bazı kurallar aşağıda belirtilmektedir.

- Bir değişken yazılırken önce tipi, sonra adı yazılır.



- Anahtar ve saklı sözcükler veri tiplerini isimlendirme sırasında kullanılamaz.
- Veri tipi isimlendirme ya bir harf ile başlamalı ya da ($) simgesi veya (_) simgesiyle başlamalıdır.
- İlk harften sonrakiler harf, rakam, ($) simgesi ya da (_) simgesi olabilir.

```
int dogrulamaKodu2;
```

- Veri tipi isimlendirmede büyük küçük harf ayrımına dikkat edilmelidir.
- Veri tipi isimlerinin içinde boşluk karakteri olamaz.
- Veri tipi isimlendirmelerinde Türkçe karakter kullanılmaz. Bazı programlama dilleri Türkçe karakterleri desteklese de çoğunluğu desteklemez. Farklı programlama dilleri ile çalışılırken sıkıntı yaşamamak için Türkçe karakterleri kullanmamak alışkanlık hâline getirilmelidir ki diğer dillerle çalışırken sıkıntı yaşanmasın.

Derleyiciler açısından değişkenlere verilen adların kısa ya da uzun olması, anlamlı ya da anlamsız olması önemli değildir. Ancak kaynak programı yazan ve okuyanın değişkenlerin neleri temsil ettiğini kolay anlayabilmesi için anlamlı isimler koymak uygun olur. Kısa programlarda bu bir sorun yaratmaz, tek harften oluşan adlar bile verilir ve çoğunlukla rahatlık sağlar. Uzun programlarda değişkenlerin temsil ettikleri veriyi belirtecek adlarla adlandırılmaları, kaynak programın kolay anlaşılır olmasını sağlar. Dolayısıyla tercih edilmesi gereken bir usuldür.

Yukarıdaki kısıtlara göre veri tiplerine verilen isimler harf, alt çizgi (_) ya da dolar ($) simgesiyle başlar. Harf, (_) ve ($) dışındaki diğer karakterlerden herhangi birisiyle başlayamaz. İsimler, ilk karakterden sonra harf, rakam, alt çizgi ve dolar simgelerinden istenilenlerle devam edebilir. Söylenenler dışındaki karakterler kullanılamaz. Örneğin artı +, -, *, “, @,), < gibi karakterler veri tipi isimleri içinde yer alamaz. Ad uzunluğu (adda kullanılacak karakter sayısı) için bir sınırlama yoktur. Varsa işletim sisteminin kısıtlarına uyulmalıdır. Örneğin DOS işletim sistemi sekiz harften uzun adları birbirinden ayıramaz.

Adlandırma kuralına uymak koşuluyla ögelere istenilen ad verebilir. Gereksiz yere uzun adlar verilmemelidir. Çünkü o öğelerle işlem yapmak için verilen adlar eksiksiz ve doğru yazılmak zorundadır.

Değişken adlarını küçük harfle, sınıf adlarını büyük harfle başlatmak bir gelenektir. Bu geleneğe uyulmalıdır.

Programda tanımlanan değişken sabit, sınıf, metot gibi varlıkların adlarını, işlevlerini ima edecek biçimde koymak program okumayı kolaylaştıran iyi bir alışkanlıktır.

Bazı durumlarda adlar birden çok sözcükten oluşabilir. O zaman sözcükler arasına alt çizgi (_) simgesi koymak izlenen yöntemlerdendir. Farklı sözcüklerin ilk harfleri büyük yazılarak veya sözcükler bitişik yazılarak da kullanmak mantıklı bir kullanım sağlayacaktır.

Aşağıda verilenler geçerli birer değişken adıdır;

- faizOrani, sicilNo, ögrencininAdiVeSoyadi, sigortalininDogumTarihi

Aynı değişkenler;

- faiz_orani, sicil_no, öğrencinin_adi_ve_soyadi, sigortalinin_dogum_tarihi diye de yazılabilir.

Aşağıda verilen kullanımlar ad olarak kullanılamaz:

- Faiz Oranı, sicil No, ad+soyad, 2.sınıf, ders-notu, Ahmet'inKitabı

Gerçek hayatta veriler sınıflandırılarak kullanılır. Sınıflandırma işlemi genellikle verilerin türlerine göre yapılır. Örneğin bir futbol liginin puan tablosunda, takımların isimleri ve puanları vardır. İsimler metinsel, puanlar ise sayısal değerlerdir. Sayıların alt alta yazılmış olması, ligde hangi takımın daha çok puana sahip olduğunu, dolayısıyla karşılaştırma yapıp büyükten küçüğe sıralanabileceğini gösterir.

Veri tipleri bir değişken için ayrılacak bölgenin hangi formatta olacağını belirlemek için kullanılır. Örneğin sayısal bir değerin saklanacağı değişken için sayısal bir veri tipi seçilir ya da metin tabanlı (adı, soyadı gibi) kavramları saklamak için metin tabanlı veri tipleri seçilir. Veri tipi belirlemede bir önemli nokta ise veri tipini seçmeden önce değişkene girilecek değerin sınırlarını belirlemektir.

Aşağıda farklı veri tiplerinin kısa açıklamaları ve alabilecekleri değerler gösterilmiştir.

Tip	Bit	Değer Aralığı
byte	8	0'dan 255'e
sbyte	8	-128'den 127'ye
short	16	-32,768'den 32,767'ye
ushort	16	0'dan 65,535'e
int	32	-2,147,483,648'den 2,147,483,647'ye
uint	32	0'dan 4,294,967,295'e
long	64	-9,223,372,036,854,775,808'den 9,223,372,036,854,775,807'ye
ulong	64	0'dan 18,446,744,073,709,551,615'e

Tablo 1.1: Tamsayılar

Tip	Bit	Değer Aralığı
float	32	1.5E-45 ile 3.4E+38
double	64	5E-324 ile 1.7E+308

decimal	128	1E-28 ile 7.9E+28

Tablo 1.2: Ondalıklı sayılar

float tipindeki bir değişkenin oluşturulma örneği aşağıdaki gibidir:

float pi = 3.14f;

decimal tipindeki bir değişken ise aşağıdaki gibi tanımlanabilir:

decimal pi = 3.14m;

"**f**" ve "**m**" harfleri, program derlenirken bu sayıların ne şekilde ele alınacağını sisteme bildirmek için kullanılır.

Tip	Bit	DeğerAralığı
char	16	Bir karakter
string	20+(n/2)*4	İstenen uzunlukta karakter dizilerini tutabilir

Tablo 1.3:Metinsel ifadeler

1.2. Metinsel Veri Tipleri

Kullanılacak olan veri tipi, bir isim, harf ya da bir karakter içeriyor ve sayısal işlem yapılmasına izin vermeyecekse bu veri tipleri kullanılır. Genellikle adı soyadı, adres ya da harf ve karakter gibi bir değişken tanımlanmasının gerektiği durumlarda kullanılan veri tipleridir.

1.2.1. Char Veri Tipi

Karakter veri tipi 2 byte bellek boyutuna ihtiyaç duymaktadır. Karakter veri tipi char, sadece bir adet harf, rakam, sembol veya alfanumerik değeri alabilir. Değer aralığı 0 ila 65.535 arasında tanımlanmış Unicode değerlerdir. Char değişkenine değer atamak için (' ') tek tırnak operatörü içine istenilen karakter yazılmaktadır. Ayrıca 'u0000' ila 'uffff' aralığında Unicode değeri verilerek de değer ataması yapılabilir.

Örnek

```
char degiskenadi ;
char cvar1;
char cvar2;
char harf = 'K' ;
```

Bildirimi char tipinden harf adlı bir değişken tanımlıyor ve ona ilk değer olarak 'K' harfini atıyor.

Burada şu ayrıntıya da dikkat edilmelidir. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 rakamlarının her biri sayısal değerlerinin dışında, ayrıca birer karakter olarak char veri tipine aittir.

Örneğin

```
char h = '7' ;
int  s =  7  ;
```

Bildirimleri birbirlerinden çok farklıdır. Birincisinde h değişkeni '7' karakterini tutan bir char değişkenidir, onunla sayısal işlemler yapılamaz. İkincisinde s değişkeni 7 sayısını tutan bir int değişkenidir, onunla sayısal işlemler yapılabilir.

Bilgisayara girdi/çıktı işlemlerinde kullanıcı ile etkileşebilmesi için sayılar '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' karakterleriyle yazılır. Girdi olurken sistem onları iki tabanlı sayıya dönüştürür; çıktı olurken iki tabanlı sayıları bu karakterlere dönüştürür.

1.2.2. String Veri Tipleri

Tek bir karakter yerine bir sözcük, bir tümce, bir paragraf ya da bir roman yazmak istediğinde char veri tipi yeterli olmayacaktır. Modern dillerin çoğunda olduğu gibi, Java dili de metinleri içerecek bir veri tipi yaratmıştır. Bu tipe String denir. String sınıfı, uzun ya da kısa her metni içerebilir. Burada metin derken yalnızca alfabenin harf, rakam ve işaretleriyle yetinilmiyor, char tiplerinden oluşabilecek her dizi kastediliyor.

String değişkenlerine atanan değerler (metinler) çift tırnak (" ") içine yazılır. Char tipler tek tırnak (' ') içine yazıldığı için Java derleyicisi 'b' verisini char olarak algılar ama "b" verisini String olarak algılar.

Örnek

```
String birMetin;
```

ya da bildirim anında ilk değeri verilmek istenirse

```
string birMetin = "Merhaba, dünya!";
```

Burada dikkat edilmesi gereken konu String değerler daima çift tırnak (" ") içinde yazıldığı için "b" , "0" , "Ankara" , "&%$*/?" , "3 + 2" , "123" , "-123" birer String'dir.

1.3. Sayısal Veri Tipleri

Birçok işlem tamsayılarla ve kesirli sayılarla yapılır. O nedenle, sayılar her programlama dilinde önemlidir. Farklı derleyiciler sayıları alt sınıflarına ayırır. En basit derleyiciler bile sayıları tamsayılar (integer) ve kesirli sayılar (floating numbers) diye ikiye

ayırır. Bazen küçük sayılarla bazen daha büyük sayılarla uğraşılır. Dolayısıyla belleği etkin kullanabilmek için sayılar alt gruplara ayrılır.

İlkel veri tipleri diye kullanılan byte, short, int, long, float ve double sayı tiplerinin hepsi soyut Number sınıfından türetilmiş Byte, Short, Integer, Long, Float, Double sınıflarına ait ögeler olarak işlev görür. Başka bir deyişle byte tipi Byte sınıfının bir ögesidir (değişken), short tipi Short sınıfının bir ögesidir vb.dir. Bu düşünüşle byte, short, int, long, float ve double ilkel sayı tiplerinin hepsi, ait oldukları sınıfa ait değişkenlermiş gibi kullanılabilir. Bu özelliğe, ilkel sayı tipinin ait olduğu sınıfa gömülmesi (wrapping) denir.

1.3.1. Byte

Byte en küçük tam sayı tipidir. Büyüklüğü 8 bit'tir ve **-128 ile +127** arası değer alır. Kod geliştirirken *byte* anahtar kelimesi ile tanımlama yapılır.

Örnek;

```
byte byteDeger = 45;
```

Yukarıda görülen kod satırında byte veri tipiyle değeri 45 olan "byteDeger" değişken adıyla bir tanımlama yapılmıştır. Yukarıda tanımlanan byteDeger değişkenine -128'den küçük veya +127'den büyük bir değer tanımlandığı takdirde geliştirme anında hata ile karşılaşılır.

1.3.2. Short

Büyüklüğü 16 bit olan short veri tipi **-32768 ile +32767** arasında bir değer alabilir. Kod geliştirirken short anahtar kelimesi ile tanımlama yapılır.

Örnek

```
short shortDeger = 64;
```

Yukarıda görülen kod satırında short veri tipiyle değeri 64 olan "shortDeger" değişken adıyla bir tanımlama yapılmıştır. Değer olarak -32769 veya 32678 verilseydi geliştirme anında hata ile karşılaşılırdı.

1.3.3. Integer

En çok kullanılan veri tipidir. 32 bitlik veri tipi, -2.147.483.648 ile 2.147.483.647 arasında bir değer alabilir. Kod geliştirirken int anahtar kelimesi ile tanımlama yapılır.

Örnek;

```
int integerDeger = 75;
```

Yukarıda görülen kod satırında integer veri tipiyle değeri 75 olan, "integerDeger" değişken adıyla bir tanımlama yapılmıştır. Yukarıda belirtilen değer aralığından daha küçük veya daha büyük bir tanımlama yapılırsa geliştirme anında hata ile karşılaşılır.

1.3.4. Long

En büyük tam sayı değeridir. 64 bitlik büyüklüğe sahiptir ve -9,223,372,036,854,775,808 ile 9,223,372,036,854,775,807 arasında bir değer alabilir. Kod geliştirirken long anahtar kelimesi ile tanımlama yapılır. Integer veri tipinin yetersiz olduğu durumlarda kullanılabilir. Bu duruma en güzel örnek bilimsel hesaplamalar veya Türkiye Cumhuriyeti Vatandaşlık Numarası tanımlamalarıdır.

Örnek

```
long longDeger = 234;
```

Bu tipleri sahip oldukları değer aralıklarına göre kullanmak çok mantıklı olmayacaktır. Bunun yerine tecrübe ettikçe hangi işlem için hangi veri tipi kullanılacağına rahatlıkla karar verilebilir. Örneğin programda kullanıcının yaşıyla işlem yapılacaksa int veya bir dosyadaki veriler okunacaksa byte veri tipini kullanmak en mantıklı yöntem olacaktır.

1.3.5. Float

İki çeşit reel sayı tipi vardır. Tam sayı tiplerinde olduğu gibi reel sayı tipleri de bitlerle ifade edilir. Float veri tipi 32 bitlik büyüklüğe sahipken double veri tipi 64 bitlik büyüklüğe sahiptir.

Float veri tipi 32 bitlik büyüklüğe sahiptir ve 1.4×10^-45 ile 3.4×10^38 aralığında bir değer tanımlanabilir. Float veri tanımlanırken değişken verisinin sonuna "f" veya "F" koyularak verinin tipinin float olduğu belirtilir. Aksi hâlde program bunu double olarak algılayacağı için hata verecektir. Aşağıdaki örneklerle konuya biraz daha açıklık getirmeye çalışalım.

Örnek

```
float floatDeger1 = 35.4f; //doğru tanım
float floatDeger2 = 4.5F; //doğru tanım
float floatDeger3 = 67; //doğru tanım
float floatDeger4 = 45.0; //hatalı tanım
float floatDeger5 = 34.65; //hatalı tanım
```

Float veri tipinin sonuna "f" veya "F" koyulmadığında program bu tanımlamadaki veri tipini double olarak algılayacağı için geliştirme anında hata verecektir. Kod geliştirirken **float** anahtar kelimesi ile tanımlama yapılır.

1.3.6. Double

Double veri tipi 64 bitlik büyüklüğe sahiptir ve 4.9×10^-324 ile 1.8×10^308 arasında bir değer tanımlanabilir. Kod geliştirirken **double** anahtar kelimesi ile tanımlama yapılır.

Örnek

```
double doubleDeger = 34.5;
```

1.4. Sabit Veri Tipleri

Sabitler, program çalışırken değer değiştiremez. Bu nedenle bildirimi yapılırken ilk değeri verilmelidir. Atanan bu ilk değer, program boyunca değişmeden kalır. Öteki değişkenlerden ayırmak için programda sabitlerin adlarını büyük harflerle yazmak bir gelenektir.

Sabitler de değişkenler gibi veri tutan sınıf öğeleridir. Dolayısıyla her sabite, ana bellekte tutacağı veri tipine yetecek kadar bir yer ayrılır. Bunun değişkenden tek farkı değişkenlere atanan değerler program çalışırken değiştirilebilir ancak sabitlere atanan değerler değiştirilemez. Ayrıca sabitlerin bildirimi yapıldığı anda değeri atanmalıdır.

Sabitler static nitelemesini aldığı için ait olduğu sınıfın bir nesnesi yaratılmadan kullanılabilir. Ana bellekte program sabitlerine bir tek yer ayrılır. Dolayısıyla ona erişen bütün ögeler aynı değeri paylaşırlar. Programda static nitelemesinin işlevi ileride daha ayrıntılı ele alınacaktır.

Sabit kullanmadan sabitin işleme girdiği her yerde onun değerini koyarak program yazmak mümkündür. Örneğin yillikFaizOrani bir bankanın mudilerinin alacağı faizleri hesaplayan programda kullanılıyor olsun. Yıllık faizin hesaplandığı her işlemde yillikFaizOrani yerine 8 kullanılabilir ve program aynı sonucu verir. Ancak faiz oranını değiştirip 9 yapmak istediğinizde kaynak programın bütününde 8 yerine 9 yazmak gerekecektir. On bin müşterisi olan bir bankada bu iş, hem uzun zaman alıcı hem de bazıları unutulabileceği için hataya açıktır. Onun yerine yillikFaizOrani adlı bir sabit kullanılırsa kaynak programda onun değerini 9 ile değiştirmek, bütün programı güncellemeye denktir.

Bunu yapmak büyük bir zaman kaybını önleyeceği gibi programda hata oluşmasının da önüne geçecektir.

1.5. Yorum Satırları

Yazılan kaynak kod içinde ilgili satır veya satırların ne iş yaptığı ile ilgili kendinize veya kaynak kodu inceleyecek kişilere bilgi verilebilir. Yazılan program derlendiğinde yorum satırları işleme alınmaz.

Java'da yorum satırlarını belirtme iki şekilde mümkündür.

- ***/* yorum */*** , slash - yıldızdan, diğer yıldız-slash arasına istenildiği kadar yorum yazılabilir. Uzun satırlı yorumlarda bu yöntem kullanabilir.

- ***// yorum***, tek satırlık yorum yapmak için idealdir. Kısa yorumlar için bu yöntem kullanabilir.

Seçili satırları hızlı bir şekilde yorum satırı hâline getirmek veya tersine çevirmek için "CTRL + SHIFT + /" kısa yolu da kullanılabilir.

DEĞER ETKİNLİĞİ - 1

Öz güven tüm insanların başarılı olabilmesinin ardındaki en önemli etkendir. Ancak biz öz güven kelimesinin anlamını tam manasıyla biliyor muyuz? Aşağıda verilen işlem basamaklarını tek tek gerçekleştirerek doğru öz güvenin ne olduğunu ve bu öz güven sayesinde nasıl başarılı olabileceğinizi öğreneceksiniz.

-Başarıdaki en önemli faktör olan kendine güvenme konusunda öğretmen kısa bir konuşma yapar.

-Öğrencilere dosya kâğıdı dağıtılır.

-Öğrencilere ileride ne olmak istedikleri hakkında sorulur.

-Öğrencilerden cevaplar alınır ve bu cevapları resmetmeleri istenir.

-Öğrencilerden resimlerini tamamladıktan sonra sırayla tahtaya kalkıp resimlerini sınıfa göstererek "Ben ……. olmayı başarabilirim. Kendime güveniyorum." demeleri istenir.

-Ayrıca ilk öğrenci hariç diğer öğrencilerin kendilerinden önceki arkadaşları için "Evet ben güveniyorum arkadaşım ……olmayı başarabilir.'' demeleri istenir.

Örneğin "Ben çok ünlü bir mühendis olabilirim ve arkadaşım Mehmet'e de güveniyorum. O doktor olmayı başarabilir."

-Konuşmalar tamamlandıktan sonra öğretmen başarının elde edilmesinde öz güvenin öneminden bahsederek etkinliği sonlandırır.

UYGULAMA FAALİYETİ-1

İş sağlığı ve güvenliği tedbirlerini alarak basit bir program yazım çalışması yaparak programda ihtiyaca uygun veri tiplerini kullanıp ve değer atamalarını yapınız.

İşlem Basamakları	Öneriler
➢ Okulunuzda kendi çalıştığınız veya evinizde bulunan android studio programı yüklü bilgisayarda yeni bir proje dosyası açınız.	➢ Bilgisayarınızda kurulu olan programın doğru çalışıp çalışmadığından emin olmalısınız.

➢ Aşağıda üç farklı uygulama bulunmaktadır. Açmış olduğunuz proje dosyasına uygulama kodlarını yazarak sonuçlarını inceleyiniz ve arkadaşlarınızın sonuçlarıyla karşılaştırınız. Doğru sonuçları bulduğunuza emin olmak için işlemlerin sağlamasını yapınız.
 	
➢ **Uygulama1**
public class uyg1 {
public static void main(String[] args) {
 	
int a, b, c; a=9; b=12; c=13;
System.out.println("a="+a);
System.out.println("b="+b);
System.out.println("c="+c);
System.out.println("a + b / c="+(a+b/c));
System.out.println("'x'in ASCII kodu="+(int)'x');
}
}
 	
➢ **Uygulama2**
public class uyg2 {
public static void main(String[] args) {
 	
int sinav1=90;
int sinav2=85;
int ort_yaklasik =(sinav1+sinav2)/2;
float ort_tam =(float) (sinav1+sinav2)/2;
 	
System.out.println("Ort. yaklaşık: "+ort_yaklasik);
System.out.println("Ort. tam: "+ort_tam);
}
}
	➢ Yazdığınız programın doğruluğundan emin olabilmek için kuralları tekrar kontrol etmelisiniz.
	
	➢ float ortalama = (float) (sinav1+sinav2)/2; satırındaki (float) ifadesi ile "(sinav1+sinav2)/2" işleminden gelen sonucun float olarak tutulması sağlanır.

➢ **Uygulama3**
public class uyg3 {
public static void main(String[] args) {
 	
float PI_float =(float) Math.PI;
double PI_double=Math.PI;
 	
System.out.println("PI sayisi: "+PI_float);
System.out.println("PI sayisi: "+PI_double);
}
}
	➢ Math.PI, PI sayısını döndürür.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatle okuyarak doğru seçeneği işaretleyiniz.

1. Aşağıdakilerden hangisi short veri tipinin değer aralığıdır?
A) -2^7……2^7-1
B) 0……..2^8-1
C) -2^{15}……$2^{15}-1$
D) 0……$2^{16}-1$
E) 0……..$2^{12}-1$

2. A integer B double C float tipinde değişkenler olsun. Aşağıdakilerden hangisi derleme hatasına yol açar?
A) double d=A*B;
B) A=B;
C) B=A;
D) B=C;
E) float E=A*C;

3. Aşağıdakilerden hangisi temel değişken türlerinden değildir?
A) int
B) decimal
C) float
D) char
E) boolean

4. Programda değişken yerine sabit kullanılmak istenirse tanımın başına aşağıdakilerden hangisini kullanmak gerekir?
A) final
B) void
C) static
D) define
E) int

5. Aşağıda verilen temel veri tipleri ve bunların boyutlarıyla ilgili olarak aşağıdakilerin hangisinde hata vardır?
A) Float 32 bit
B) Long 64 bit
C) Byte 8 bit
D) Double 32 bit
E) Short 16 bit

6. ……. değişken türü harflerin tanımlanmasında kullanılır. Boş bırakılan yere aşağıdakilerden hangisi gelmelidir?
A) String
B) New string
C) Hiçbiri
D) Char
E) String-char

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–2

ÖĞRENME KAZANIMI

İşlem önceliklerini dikkate alarak operatörleri doğru bir şekilde kullanabileceksiniz.

ARAŞTIRMA

- Farklı programlama dillerinde kullanılan operatörlerin neler olduğunu ve kullanım şekillerini araştırınız.
- Daha önce kullanmış olduğunuz operatör şekillerinin bu programlama dilindeki kullanım şekliyle birlikte küçük çapta bir program yazarak sınıfınızla paylaşınız.

2. OPERATÖRLER VE KULLANIMLARI

Programlama dillerinde tanımlanmış sabit ve değişkenler üzerinde işlemler yapmayı sağlayan karakter ya da karakter topluluklarına **operatör** denir.

2.1. Tekli Operatörler

Tekli operatörler bir değişkenin sağına veya soluna gelerek tek başına değişkenin değerini değiştirebilen operatörlerdir. + (artı), - (eksi), ++ (1 arttırma), -- (1 azaltma), ! (boolean tipi tersine çevirme) operatörleri mevcuttur.

Bu operatörleri pekiştirmek için aşağıdaki örnek verilmiştir.

```
public class Uyg1 {
   public static void main(String[] args){
      int x, y=-5;

      x=16- -y;
      System.out.println("x= "+x);
   }
}
```

Bu programda **x**'in değeri 11 olarak bulunacaktır. **–y – (–5) = +5** olarak hesaplanır. 16'yı izleyen çıkarma operatörü (**–**) ile y'nin önündeki negatifini alma (**–**) operatörü arasında mutlaka bir boşluk olmalıdır. Aksi hâlde (**–)** ifadesi azaltma operatörü olarak algılanacaktır.

tekli negatif alma	–	–a	a'nın negatifini alır
son artım	++	a++	önce a'nın değerini al, kullan, sonra 1 artır.
ön artım	++	++a	önce a'nın değerini 1 artır, sonra kullan.
son azaltım	– –	a– –	önce a'nın değerini al, kullan, sonra 1 son azaltım azalt.
ön azaltım	– –	– –a	önce a'nın değerini 1 azalt, sonra kullan.

Tablo 2.1: Tekli operatörler

Son artım ve son azaltım operatörlerinde derleyici, değişkenin o andaki değerini bellekte geçici bir değişkene atayarak saklar. Sonra değişkenin değerini 1 artırır (son artım) veya 1 azaltır (son azaltım). Geçici değişkendeki değer ise ifade içinde kullanılır.

2.2. İkili Aritmetiksel Operatörler

Aritmetik operatörler klasik matematiksel işlemlerin yapılmasını sağlayan operatörlerdir. + (toplama), - (çıkarma), *(çarpma), /(bölme) ve %(mod alma) işlemlerinin yapılmasını sağlar.

Operatör	Sembolü	Kullanılışı	İşlem sonucu
Çarpma	*	a*b	a ile b'nin çarpımı
Bölme	/	a/b	a'nın b'ye bölümü
Kalan	%	a%b	a'nın b'ye bölümünden kalan
Toplama	+	a+b	a'nın b ile toplanması
Çıkarma	–	a-b	b'nin a'dan çıkarılması

Tablo 2.2: İkili aritmetiksel operatörler

***, /** ve % operatörleri **+** ve **–** ' ye göre önceliklidir. ***, /, +** ve **–** operatörlerinin int ve ya float (double) türde operand kabul etmelerine karşılık kalan operandları sadece int türde operand olarak kabul eder. % operandı bölmede kalanı hesaplar.

Örnek: 17 % 3 işleminin sonucunda 2 değeri (17'nin 3 ile bölümünden kalan) elde edilir. ANSI standardı, kalan ve bölme operatörleri arasında aşağıdaki bağıntının bulunmasını zorunlu kılar:

a = a % b + (a /b) *b (a ve b tamsayı)

2.3. Aritmetiksel Atama Operatörleri

Bu operatörler değişkenlere değer atama ve aritmetik işlemlerin tek bir defada yapılmasını sağlar.

2.3.1. Topla ve Ata Operatörü (+=)

Solundaki değişkenin değerini sağındaki değişkenin değeri kadar artırır ve sonucu atar.

```
int sayi1 = 45;
int sayi2 = 25;
sayi1 += sayi2;              //Sonuç = 70
```

2.3.2. Çıkart ve Ata Operatörü (-=)

Solundaki değişkenin değerini sağındaki değişkenin değeri kadar azaltır ve sonucu atar.

```
+int sayi1 = 45;
int sayi2 = 25;
sayi1 -= sayi2;       //Sonuç = 20
```

2.3.3. Çarp ve Ata Operatörü (*=)

Solundaki değişkenin değerini sağındaki değişkenin değeriyle çarpar ve sonucu atar.

```
int sayi1 = 5;
int sayi2 = 2;
sayi1 *= sayi2;       //Sonuç = 10
```

2.3.4. Böl ve Ata Operatörü (/=)

Solundaki değişkenin değerini sağındaki değişkenin değerine böler ve sonucu atar.

```
int sayi1 = 45;
int sayi2 = 5;
sayi1 /= sayi2;       //Sonuç = 9
```

2.3.5. Mod ve Ata Operatörü (/=)

Solundaki değişkenin değerini sağındaki değişkenin değerine göre modunu alır ve sonucu atar.

```
int sayi1 = 45;
int sayi2 = 25;
sayi1 %=sayi2;        //Sonuç = 20
```

2.4. Mantıksal Operatörler

Mantıksal operatörler birden çok karşılaştırma işlemini birleştirip tek bir koşul ifadesi hâline getirmek için kullanılır.

X	Y	X AND Y	X OR Y	X XOR Y	Not X veya (!X)
false	false	false	false	false	true
false	true	false	true	true	true
true	false	false	true	true	false
true	true	true	true	false	false

Tablo 2.3. Mantıksal operatörler

2.4.2. AND Operatörü

Bu operatör "&&" şeklinde de kullanılabilir. Üstteki tabloda da (Tablo 2.3) görüldüğü gibi koşullardan biri yanlış (false) olduğunda yanlış sonucunu verir. İki koşulda doğru (true) olduğunda, doğru sonucunu verir.

2.4.3. OR Operatörü

Bu operatör "||" şeklinde de kullanılabilir. Üstteki tabloda da (Tablo 2.3) görüldüğü gibi koşullardan en az biri doğru (true) olduğunda doğru sonucunu verir. İki koşulda yanlış (false) olduğunda yanlış sonucunu verir.

2.4.4. NOT Operatörü

Bu operatör "!" ifadesi ile kullanılabilir. Bu operatör kendisinden sonra gelen mantıksal ifadenin değilini alır.

2.4.5. XOR Operatörü

Her iki ifade doğru veya yanlış ise yanlış, birisi doğru birisi yanlış ise doğru sonucunu verir.

Örnekler

```
int sayi1=10;
int sayi2=5;
boolean c=true;

a > b && b > 2 ;  //Sonuç = true
a > b & b > 2;            //Sonuç = true
a > b && b > 7;           //Sonuç = false
a > b || b > 7;           //Sonuç = true
```

```
b > a || b > 2 ;        //Sonuç = true
a > b ^ b > 2 ;         //Sonuç = false
a > b ^ b > 7 ;         //Sonuç = true
b > a ^ b > 2;          //Sonuç = true
b > a ^ b > 7;          //Sonuç = false
```

Mantıksal operatörlerden && (AND), || (OR) operatörleri tek karakter (& , |) olarak da kullanılabilir.

Aralarındaki fark ise tek karakterli operatörlerin her iki yanındaki koşulları kesinlikle yoklamasıdır. Çift karakterli işleme soldan başlanır ve tüm ifade bitmeden kesin sonuca ulaşılırsa ifadenin geri kalan kısmı göz ardı edilebilir. Örneğin AND işleminde sol taraf yanlış ise sonuç kesin yanlış olacaktır. Çift karakterli (&&) operatör kullanılırsa burada ilk koşul uymadığı için ikinci koşula bakılmaz ama tek karakterli (&) operatör kullanılırsa birinci koşul yanlış olsa da ikinci koşul da kontrol edilir.

2.5. Operatör Önceliği

Matematiksel işlemlerde kullanılan operatörlerin işlem önceliği vardır. Bu öncelik sırası matematiktekinin aynısıdır.

Aşağıdaki tabloda operatörlerin öncelik sırası gösterilmektedir.

Açıklama	Operatör	Öncelik	Okunuş
Parantez	( )	11	Soldan Sağa
Sonra artır **Sonra eksilt** **Nokta**	++ – .	10	Sağdan Sola
Önce Artır **Önce Eksilt** **Tekil Operatörler**	++ – ! –	9	Soldan Sağa
Tip Atama Nesne Oluşturma	(type) new	8	Soldan Sağa
Çarpma-Bölme	*/%	7	Soldan Sağa
Toplama **Çıkarma** **String Toplama**	+ – +	6	Soldan Sağa
İlişkisel Operatörler	< <= > =>	5	Soldan Sağa
Eşitlik İnceleme	==	4	Soldan Sağa

	!=		
Mantıksal XOR	^	3	Soldan Sağa
Mantıksal AND	&&	2	Soldan Sağa
Mantıksal OR	||	1	Soldan Sağa
Atama Operatörleri	=	0	Sağdan Sola

Tablo 2.4. Operatör öncelikleri

Önceliği en yüksek olan operatör ilk çalışacak operatördür.

DEĞERLER ETKİNLİĞİ-2

Üç öğrenci seçilerek aşağıdaki kişilerin hayatlarını bir diğer derse kadar araştırarak panel yapmaları istenir.

1-Sakıp Sabancı'nın hayat mücadelesi
2-Hz. Ali'nin Müslüman oluşu
3-Mahatma Gandi'nin özgürlük mücadelesi

Öğrenciler bu kişiler hakkındaki buldukları bilgilerle panel yaptıktan sonra sınıftan bu konulardan hareketle sabır, azim, dayanıklılık, beklemeyi bilme gibi temalarda yazı yazınız.
Sınıfta seçilen yazı panoya asılır.

UYGULAMA FAALİYETİ-2

İş sağlığı ve güvenliği tedbirlerini alarak aşağıdaki işlem basamaklarını takip ederek operatörlerin kullanımını inceleyiniz**.**

İşlem Basamakları	Öneriler

➢ Bilgisayar laboratuvarında sizin kullanımınıza ayrılmış olan bir bilgisayarda android studio programını kullanarak tekli operatörlerle ilgili aşağıdaki programı yazınız ve sonuçlarını kontrol ediniz.
 	
public class uyg1 {
public static void main(String[] args){
int x=5, y=3, z=2;
 	
z= x++ + y--;
 	
System.out.println("z="+z);
System.out.println("x="+x);
System.out.println("y="+y);
}
}
	➢ Bilgisayarınızda kurulu olan programın doğru çalışıp çalışmadığından emin olmalısınız.

Bilgisayar laboratuvarında sizin kullanımınıza ayrılmış olan bir bilgisayarda android studio programını kullanarak ikili aritmetiksel operatörlerle ilgili aşağıdaki programı yazınız ve sonuçlarını kontrol ediniz.
 	
public class uyg2 {
public static void main(String[] args){
int a, b, c;
double p, q, r, x, y, z, k;
 	
a=6; b=7; c=12;
p=2.7;
q=3.14;
r=5.12;
 	
x= a / b * 15 - c / b - a;
y= a * 2 % (b + 1) - c / (a +b);
z=x / y + a - b ++ * --c;
k=p -q / (r +z) / (q -r) + b++ * z;
 	
System.out.println("x="+x);
System.out.println("y="+y);
System.out.println("z="+z);
System.out.println("k="+k);
}
}
	➢ Programın tüm adımlarında yazılım kurallarına uyduğunuzdan emin olmalısınız.

➢ Bilgisayar laboratuvarında sizin kullanımınıza ayrılmış olan bir bilgisayarda android studio	➢ Programın tüm adımlarında yazılım kurallarına uyduğunuzdan emin

programını kullanarak aritmetiksel operatörlerle ilgili aşağıdaki programı yazınız ve sonuçlarını kontrol ediniz.
 	
public class uyg3{
public static void main(String[ ] args){
 	
int k;
double l;
l = k =(int)13.72;
k += 4;
l -= k;
 	
System.out.println("l="+l);
System.out.println("k="+k);
}
}
	olmalısınız.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatle okuyarak doğru seçeneği işaretleyiniz.

1. ++ sembolünün anlamı aşağıdakilerden hangisidir?
A) Sayıyı her defasında arttırır.
B) Sayıya 1 ilave eder.
C) Sayıyı baştan 1 arttırır.
D) Sayıya işlem bitiminde 1 arttırır.
E) Hiçbiri

2. Aşağıdakilerden hangisi mantıksal işlem operatörlerinden değildir?
A) &&
B) <
C) !
D) <=
E) =

3. int a=2; int c; c=a++; işleminden sonraki durum aşağıdakilerden hangisidir?
A) a'nın değeri 3, c'nin değeri 3 olur.
B) a'nın değeri 2, c'nin değeri 3 olur,
C) a'nın değeri 3, c'nin değeri 4 olur,
D) a'nın değeri 3, c'nin değeri 2 olur,
E) a'nın değeri 2, c'nin değeri 2 olur,

4. += işlemi …….. taraftaki sayıyı eklemek için kullanılır. Cümlesinde boş bırakılan yere aşağıdakilerden hangisi getirilmelidir?
A) sol
B) sağ
C) Toplama
D) Alt
E) Üst

Aşağıdaki soruları dikkatle okuyunuz ve önce programı yazıp bilgisayarınızda oluşan ekran çıktılarına göre cevapları yazınız.

int a=8; int b=13; int x= a+b; verilen bir program için aşağıdaki işlemleri yapıp sonuçlarını yazınız.

5. double c=a*b işlemini yaptıran programı yazıp ekran çıktısındaki c değerini yazınız.
6. float d=a/b işlemini yaptıran programı yazıp ekran çıktısındaki c değerini yazınız.
7. int e=a/b işlemini yaptıran programı yazıp ekran çıktısındaki c değerini yazınız.
8. int f=a++ işlemini yaptıran programı yazıp ekran çıktısındaki c değerini yazınız.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise "Modül Değerlendirme"ye geçiniz.

MODÜL DEĞERLENDİRME

Bu materyal kapsamında aşağıda listelenen davranışlardan kazandığınız becerileri, **EVET**, kazanamadığınız becerileri **HAYIR** kutucuğuna (X) işareti koyarak kendinizi değerlendiriniz.

Değerlendirme Ölçütleri	Evet	Hayır
1. Bilgisayarınızda android studio programının kurulu olup olmadığını kontrol ettiniz mi?		
2. Program kurulu ise doğru çalışıp çalışmadığından emin oldunuz mu?		
3. Tüm öğrenme faaliyetlerini kapsayan bir program yazmayı denediniz mi?		
4. Yazdığınız programda nerelerde hata yaptığınızı fark ettiniz mi?		
5. Değişken tanımlamada hata yaptınız mı?		
6. Aritmetiksel operatörleri kullanırken hata yaptınız mı?		
7. Mantıksal operatörleri kullanırken hata yaptınız mı?		

DEĞERLENDİRME

Değerlendirme sonunda "**Hayır**" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "**Evet**" ise bir sonraki bireysel öğrenme materyaline geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ 1'İN CEVAP ANAHTARI

1	**C**
2	**C**
3	**B**
4	**C**
5	**D**
6	**E**

ÖĞRENME FAALİYETİ 2'NİN CEVAP ANAHTARI

1	**D**
2	**E**
3	**B**
4	**B**
5	**21**
6	**54**
7	**0,615384615384**
8	**hata**
9	**9**

KAYNAKÇA

- ARTAR Çağlar, **Java ile Android Programlama**, 1.Baskı, **,** Dikeyeksen Yayıncılık, Kasım 2015.
- https://www.baskent.edu.tr/~tkaracay/etudio/ders/prg/java/ch05/dataTypes.htm (10.07.2017/ 11.40)
- www.baskent.edu.tr/~tkaracay/etudio/ders/prg/c/dataTypes.pdf (14.07.2017/14.55)