

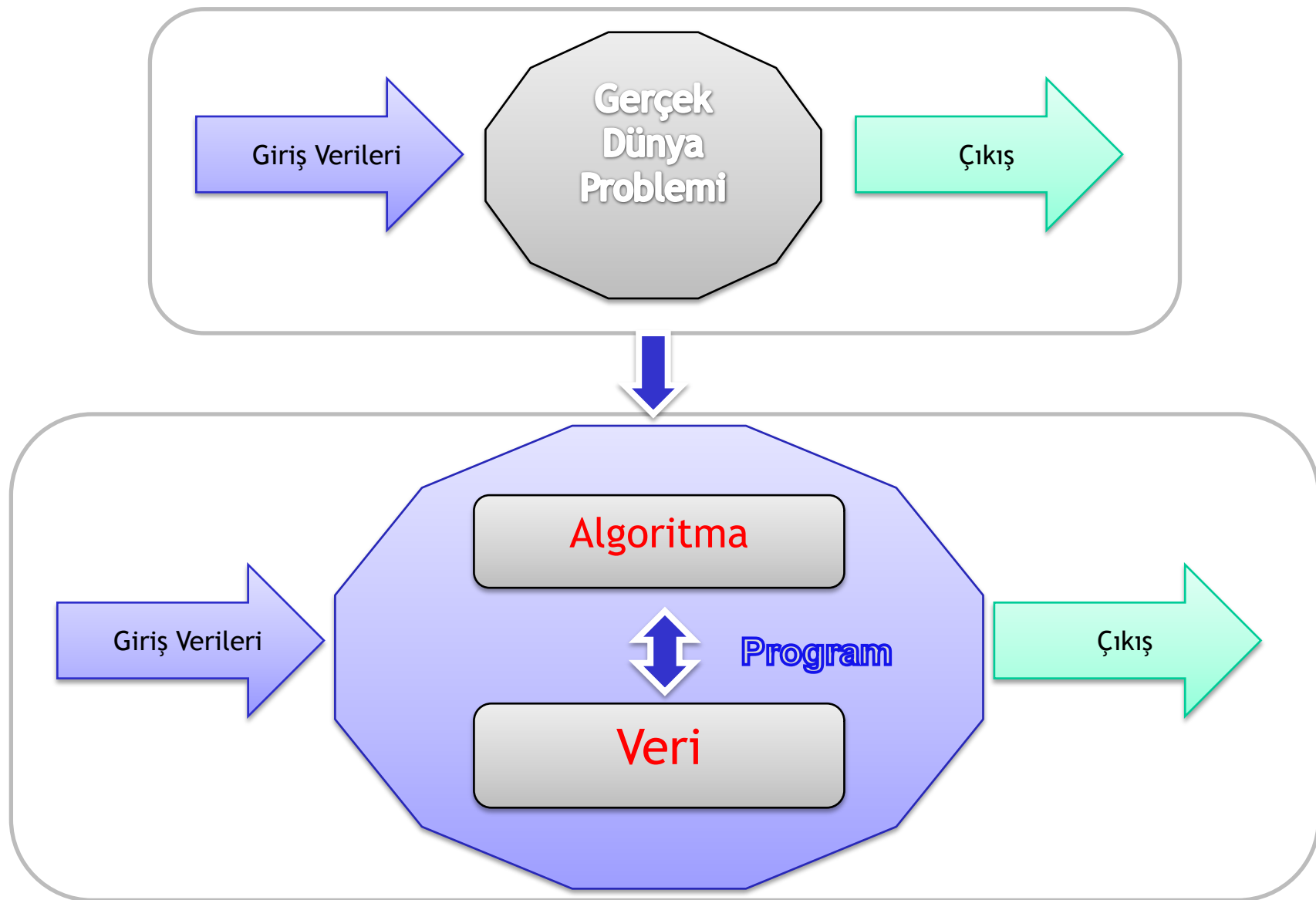
Algoritma ve Programlama - II



Konular

- ✓ Temel Kavramlar
- ✓ Yazılım Geliştirme
- ✓ C++ Ortamı Bileşenleri
- ✓ C++ Programının Gelişim Aşamaları
- ✓ İlk C++ Programı
- ✓ İsim Uzayı (Namespace)
- ✓ Değişkenler (Variables)
- ✓ Setw (Manipulator)
- ✓ Tip Dönüşümü
- ✓ İlişkisel İşleçler
- ✓ İşlem Önceliği
- ✓ Aritmetik Atama, Artırma, Azaltma İşleçleri
- ✓ Kitaplık Fonksiyonları
- ✓ Sorular

Algoritmik Problem



Temel Kavramlar

- ✓ **Algoritma:** Bir problemin çözümüne yönelik işlem basamaklarının sıralı olarak ifade edilmesi
- ✓ **Program:** Probleme ilişkin çözümlerin, herhangi bir programlama dili kullanarak, bilgisayarın anlayacağı komutlar dizisi şeklinde yazılmasına **program**, bilgisayarda kullanılan programların genel adına da **yazılım** denir.
- ✓ Ders boyunca; herhangi bir gerçek dünya problemini bilgisayar yardımıyla çözebilecek **algoritmaların nesne yönelimli olarak geliştirilmesi** anlatılacaktır.

Yazılım Geliştirme Aşamaları

Çözümleme (Analiz): Belirsizlik kalmayacak şekilde çözülmesi istenen problem anlaşılır ve yazılımın gerçekleştirilmesine yönelik hazırlıklar yapılır. İhtiyaç listesi oluşturulur...

Tasarım: Bu aşamada amaç, gerçek dünyadaki problemin bilgisayarda temsil edilebilecek soyut bir modelinin oluşturulmasıdır. Çözümün hangi unsurlardan oluşacağı ve bu parçaların nasıl modelleneneceği tamamıyla programlama yöntemine bağlıdır.

Tasarım sonrası ortaya çıkacak olan yazılımın kalitesini doğrudan etkilediğinden bu aşamada iyi ve doğru bir çözümün oluşturulması çok önemlidir. Bu nedenle kodlamaya geçilmeden önce kurulan modelin sağlamlasının (verification) yapılması gerekir.

Kodlama: Tasarım aşamasında oluşturulan model, bir programlama dili ile bilgisayara bu aşamada aktarılır.

Test: En son aşamada programın olası giriş değerleri için nasıl davrandığı incelenir. Özellikle kritik değerler için programın çalışması test edilmelidir.

Hata Ayıklama: (Yazım Hatası, Mantıksal Hata, Çalışma Zamanı Hatası)

Dokümantasyon: Yazılım projesinin her aşamasında yapılan işleri açıklayan dokümanlar hazırlamak gereklidir.

Bir Yazılımın Kalitesi

Etkinlik Bu konu algoritmaların karşılaştırılmalarında kullanılan ölçütlerle aynıdır. Program ne kadar hızlı çalışıyor? Düzgün kullanılabilmesi için ne kadar sistem kaynağı gerekiyor (hangi işlemci, ne kadar bellek, ne kadar disk, v.b.?).

Sağlamlık Program, kullanıcının yapacağı kullanım hatalarına karşı dayanıklı mı? Sözgelimi, kendisine bir yıl bilgisi sorulduğunda kullanıcı yanlışlıkla (ya da kötü niyetle) bir isim yazarsa ne oluyor?

Taşınabilirlik Program, üzerinde fazla değişiklik yapılması gerekmeden, başka bir donanım ve başka bir işletim sistemi üzerinde çalışacak hale getirilebiliyor mu?

Anlaşılabilirlik Başka biri programı okuduğunda anlayabilecek mi? Hatta üstünden bir süre geçtikten sonra kendiniz baktığınızda anlayabilecek misiniz?

Bakım kolaylığı Programda hata olduğunda hatanın kaynağının belirlenmesi ve düzeltilmesi kolay mı?

Geliştirilebilirlik Program yeni yeteneklerin eklenmesiyle geliştirilmeye açık mı? Bu tip eklemelerin yapılması kolay mı?

Nesneye dayalı programlama yukarıdaki maddelerde belirtilen kalite kriterlerini sağlamak üzere ortaya konmuş bir yöntemdir.

Programlama Dilleri

1. Makine Dili (Düşük Seviye)

Anlaşılması zor, ikili (ya da hex) sayılardan oluşur, makine bağımlıdır

```
+1300042774  
+1400593419  
+1200274027
```

2. Assmbly Dili (Orta Seviye)

Anlaşılması biraz daha kolay, ingilizce kısaltmalardan oluşur, makine bağımlıdır

Assembler derleyicisi ile makine koduna dönüştürülür.

```
LOAD sayi1
```

```
ADD sayi2
```

```
STORE toplam
```

3. Yüksek Seviyeli Diller (High Level Language)

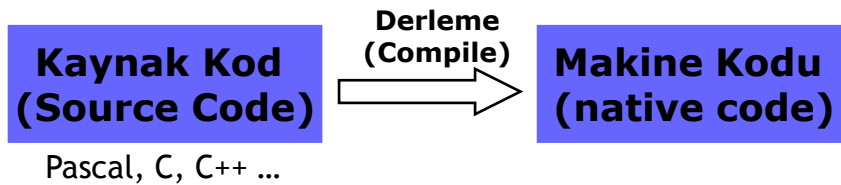
Anlaşılması çok daha kolay, komutlar ingilizce ve matematiksel ifadelerden oluşur,

Derleyici ile makine koduna dönüştürülür.

```
toplam = sayi1 + sayi 2
```

Programların Çalıştırılması

- ✓ Derleme (Compiled Language)



- ✓ Yorumlama (Interpreted Language)

Komutlar teker teker çalıştırılır

Komut okuma ve çevirme işlemi çalışma zamanında yapılır (düşük hız)

Hata düzeltme daha kolaydır.

Derleyiciden kaynaklanan sınırlamalar kalktığı için daha esnek bir çalışma ortamı.

Perl, tcl, basic, matlab...

- ✓ Karma (Hybrid)

Kodlar derlenerek byte code oluşturulur.

Byte code yorumlanarak çalıştırılır.

Java, phyton

Programlama Yöntemleri

- ✓ Yapısal olmayan programlama tekniği
- ✓ Prosedurel programlama tekniği
- ✓ Yapısal (modüler) programlama tekniği
- ✓ Nesne Yönelimli programlama tekniği

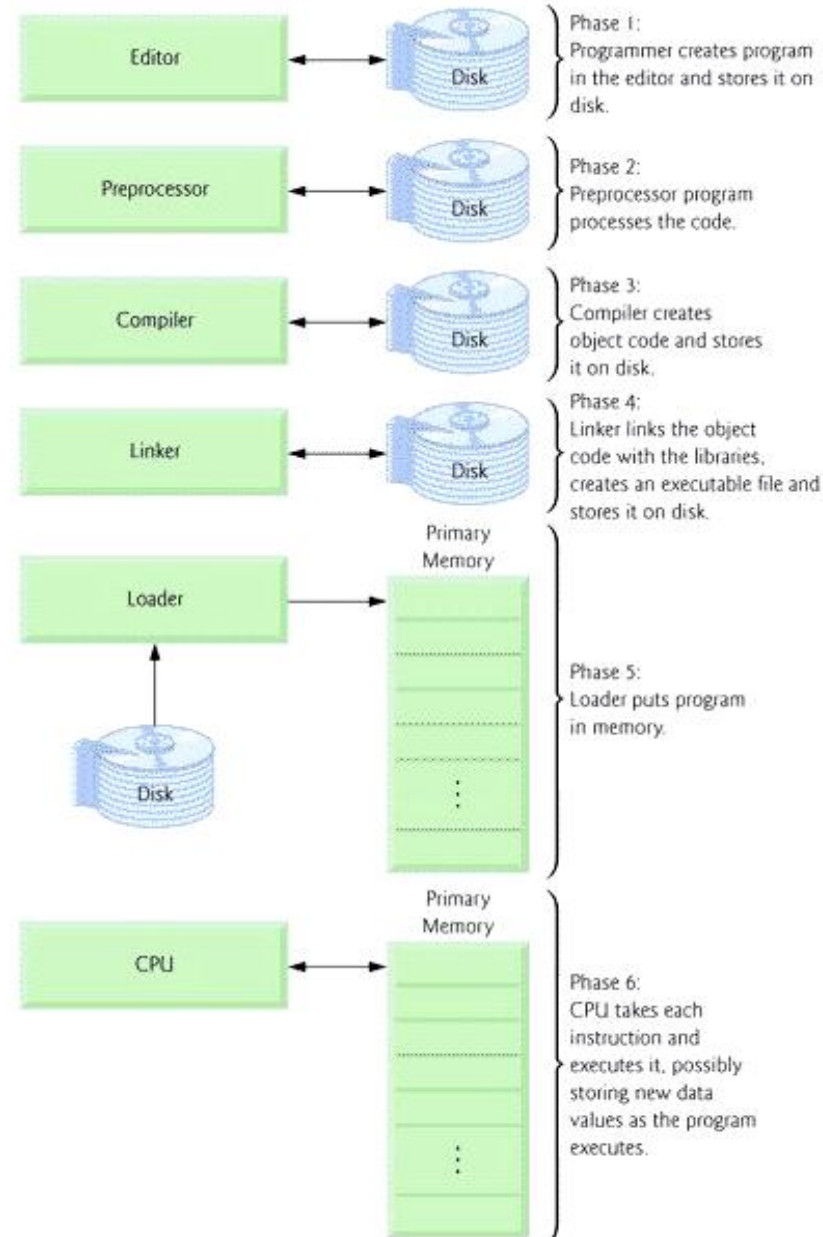
C++ Ortamı Bileşenleri

- ✓ Program Geliştirme Ortamı
- ✓ C++ Dili
- ✓ C++ Standart Kitaplığı

C++ Programının Gelişim Aşamaları

C++ Programının Gelişim Aşamaları

1. Edit
2. Preprocess
3. Compile
4. Link
5. Load
6. Execute



İlk C++ Programı

```
//=====
// Name      : AlgoritmaveProgramlama211.cpp
// Author     : Celal Çeken
// Version    :
// Copyright   : Your copyright notice
// Description : Ekrana "Merhaba Dünya" yazan program
//=====

#include <iostream>
using namespace std;

int main()
{
    cout << "!!!Merhaba Dünya!!!"<< endl; // Merhaba Dünya
    return 0;
}
```

- ✓ Anlaşılabilirliği artırmak için programların ilk satırlarında programın kısa tanıtımı, yazarı, tarihi gibi bilgiler yer almalıdır.
- ✓ Boşluklar, boş satırlar ve hiyerarşik hizalama yine anlaşılabilirliği artırma açısından son derece önemlidir.

İlk Programın Çalıştırılması

- ✓ Derleyici (GNU C Compiler (windows sistemler için mingw))
- ✓ Geliştirme ortamı (NotPad ++, Eclipse CDT, DevC++)
- ✓ makefile dosyasının oluşturulması

İsim Uzayı (Namespace)

```
namespace programci1{ //programci1'in isim uzayı
int tamsayi; // programci1'e ait tamsayi
void f(int);
: // Diğer değişkenler
} // İsim uzayının sonu
namespace programci2{ // programci2'nin isim uzayı
int tamsayi; // programci2'ye ait tamsayi
: // Diğer değişkenler
} // İsim uzayının sonu
```

İsim Uzayı (Namespace)

İsim uzaylarındaki tanımlayıcılara erişilmesi:

```
programci1::tamsayi =3; //progamci1'in uzayındaki tamsayi  
programci2::tamsayi=-345; //progamci2'nin uzayındaki tamsayi  
programci1::f(6); //progamci1'in uzayındaki f fonksiyonu
```

using Bildirimi:

```
using programci1::tamsayi; // İsim uzayındaki bir değişken için geçerlidir  
tamsayi = 3; // programci1'deki tamsayi  
programci2::tamsayi = -345; // Diğer tamsayi için isim uzayını belirtmek  
gerekli  
programci1::f(6); // f fonksiyonu için isim uzayını belirtmek gerekli
```

Bu bildirim istenirse tek bir veri yerine isim uzayının tamamı için geçerli yapılabilir. Bunun için using namespace sözcükleri birlikte kullanılır.

```
using namespace programci1; // İsim uzayının tamamı için geçerli  
tamsayi = 3; // programci1'deki tamsayi  
f(6); // programci1'deki f  
programci2::tamsayi = -345; // Diğer isim uzayı için isim belirtmek gerekli
```

Değişkenler (Variables)

Üzerinde işlem yapılan verinin saklandığı bellek konumunu gösterir.
Aşağıdaki özelliklere sahiptir.

Ad (name)

harf, rakam ve _
rakam ile başlayamaz
büyük/küçük harf duyarlı

Tip (type)

Değer (value)

Ömür (lifetime)

Değişkenin kullanılabildiği süre

İsim Uzayı (namespace)

Değişkene ismi ile erişilebilen program bloğu

Değişkenler

Değişken isimlendirme kuralları

- ✓ Değişken isimleri harf, rakam ve '_' ifadelerinden oluşur.
- ✓ İlk karakter rakam olamaz. Türkçe karakterler kullanılamaz.
- ✓ Ayrılmış kelimeler (reserved words) kullanılamaz. If, else, for v.s.
- ✓ Büyük harf-küçük harf duyarlılığı vardır (case-sensitive).
- ✓ Değişken isminin içerdiği veri ile ilgili olması büyük kolaylıklar sağlar.

ogrenciSayisi

- ✓ Değişkenlerin ilk harfi küçük, diğer kelimelerin baş harfi büyük olmalı.

ogrencininNotOrtalamasi gibi...

Kodlama türleri

Linux Coding Style, Linus Torvalds

Hungarian Notation,

MSDN GNU Coding Standards

Java Coding Style Guide

Değişken Tipleri

DATA TYPES

| Name | Bytes* | Description | Range* |
|--------------------|--------|--|--|
| char | 1 | character or integer 8 bits length. | signed: -128 to 127 unsigned: 0 to 255 |
| short | 2 | integer 16 bits length. | signed: -32768 to 32767 unsigned: 0 to 65535 |
| long | 4 | integer 32 bits length. | signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295 |
| int | * | Integer. Its length traditionally depends on the length of the system's Word type , thus in MSDOS it is 16 bits long, whereas in 32 bit systems (like Windows 9x/2000/NT and systems that work under protected mode in x86 systems) it is 32 bits long (4 bytes). | See short , long |
| float | 4 | floating point number. | 3.4e +/- 38 (7 digits) |
| double | 8 | double precision floating point number. | 1.7e +/- 308 (15 digits) |
| long double | 10 | long double precision floating point number. | 1.2e +/- 4932 (19 digits) |
| bool | 1 | Boolean value. It can take one of two values: true or false NOTE: this is a type recently added by the ANSI-C++ standard. Not all compilers support it. Consult section bool type for compatibility information. | true or false |
| wchar_t | 2 | Wide character. It is designed as a type to store international characters of a two-byte character set. NOTE: this is a type recently added by the ANSI-C++ standard. Not all compilers support it. | wide characters |

int_8, _int16, _int32, _int32, _int64

int tipi short tipinden büyük olmasına karşın daha hızlı erişilir.

Integer

```
// intvars.cpp
// demonstrates integer variables
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
    int var1;          //define var1
    int var2;          //define var2

    var1 = 20;          //assign value to var1
    var2 = var1 + 10;    //assign value to var2
    cout << "var1+10 un degeri"<<endl<<var2; //output text
    // cout << var2 << endl; //output value of var2
    char ch=getch();
    return 0;
}
```

İmleçin bir sonraki satıra geçmesini sağlar.. Std içerisinde tanımlanmıştır. std::endl

Char

```
// charvars.cpp
// demonstrates character variables
#include <iostream>      //for cout, etc.
#include <conio.h>
using namespace std;

int main()
{
    char charvar1 = 'A'; //define char variable as character
    char charvar2 = '\t'; //define char variable as tab
    cout << charvar1;    //display character
    cout << charvar2;    //display character
    charvar1 = 'B';      //set char variable to char constant
    cout << charvar1 ;    //display character
    cout << '\n';        //display newline character
    char ch=getch();
    return 0;
}
```

```
cout << "Öğretmen\"dersimiz matematik \" dedi
```

Char

| Escape Sequence | Description |
|-----------------|--|
| <code>\n</code> | Newline. Position the screen cursor to the beginning of the next line. |
| <code>\t</code> | Horizontal tab. Move the screen cursor to the next tab stop. |
| <code>\r</code> | Carriage return. Position the screen cursor to the beginning of the current line; do not advance to the next line. |
| <code>\a</code> | Alert. Sound the system bell. |
| <code>\\</code> | Backslash. Used to print a backslash character. |
| <code>\"</code> | Double quote. Used to print a double quote character. |

Cin

```
// fahrenheit.cpp
// demonstrates cin, newline
#include <iostream>
using namespace std;

int main()
{
    int ftemp; //for temperature in fahrenheit

    cout << "Enter temperature in fahrenheit: ";
    cin >> ftemp;
    int ctemp = (ftemp-32) * 5 / 9;
    cout << "Equivalent in Celsius is: " << ctemp << '\n';
    return 0;
}
```

Float

```
// circarea.cpp
// demonstrates floating point variables
#include <iostream>           //for cout, etc.
using namespace std;

int main()
{
    float rad;                //variable of type float
    const float PI = 3.14159F; //type const float

    cout << "Dairenin yari capi: "; //prompt
    cin >> rad;                       //get radius
    float alan = PI * rad * rad;      //find area
    cout << "alan degeri: " << alan << endl; //display answer
    return 0;
}
```

Kayan nokta sayılar üstel de yazılabilir: 1.55E12

Setw Sabiti (Manipulator)

```
// width1.cpp
// demonstrates need for setw manipulator
#include <iostream>
using namespace std;

int main()
{
    long pop1=2425785, pop2=47, pop3=9761;

    cout << "LOCATION " << "POP." << endl
         << "Portcity " << pop1 << endl
         << "Hightown " << pop2 << endl
         << "Lowville " << pop3 << endl;
    return 0;
}
```


Setw Sabiti (Manipulator)

```
// width2.cpp
// demonstrates setw manipulator
#include <iostream>
#include <iomanip>    // for setw
using namespace std;

int main()
{
    long pop1=2425785, pop2=47, pop3=9761;

    cout << setw(8) << "LOCATION" << setw(12)
         << "POPULATION" << endl
         << setw(8) << "Portcity" << setw(12) << pop1 << endl
         << setw(8) << "Hightown" << setw(12) << pop2 << endl
         << setw(8) << "Lowville" << setw(12) << pop3 << endl;
    return 0;
}
```

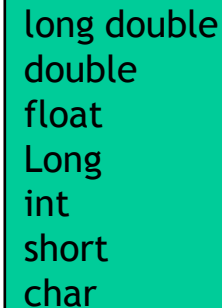
Tip Dönüşümü (Otomatik)

```
// mixed.cpp
// shows mixed expressions
#include <iostream>
using namespace std;

int main()
{
    int count = 7;
    float avgWeight = 155.5F;

    double totalWeight = count * avgWeight;
    cout << "totalWeight=" << totalWeight << endl;
    return 0;
}
```

Tiplerin Dönüşüm Sırası



long double
double
float
Long
int
short
char

Tip Dönüşümü (Otomatik)

long double

double

float

unsigned long int (synonymous with unsigned long)

long int (synonymous with long)

unsigned int (synonymous with unsigned)

int

unsigned short int (synonymous with unsigned short)

short int (synonymous with short)

unsigned char

char

bool

Tip Dönüşümü (Açık)

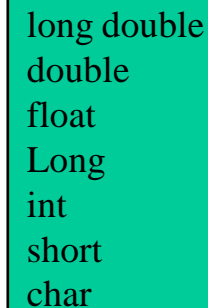
```
#include <iostream>
#include <conio.h>
using namespace std;
```

```
int main()
{
    char ch1=getch();

    cout << ch1 <<"nin ASCII degeri="<<static_cast<int>(ch1)<<
endl;
    char ch=getch();
    return 0;
}
```

The cast operator can be used to convert between fundamental numeric types, such as int and double, and between related class types (as we discuss in Chapter 13, Object-Oriented Programming: Polymorphism). Casting to the wrong type may cause compilation errors or runtime errors

Tiplerin Dönüşüm Sırası



```
long double
double
float
Long
int
short
char
```

İlişkisel İşleçler

| Standard algebraic equality operator or relational operator | C++ equality or relational operator | Example of C++ condition | Meaning of C++ condition |
|---|-------------------------------------|--------------------------|---|
| <i>Relational operators</i> | | | |
| > | > | x > y | x is greater than y |
| < | < | x < y | x is less than y |
| ≥ | >= | x >= y | x is greater than or equal to y |
| | | | |

İlişkisel İşleçler

```
// relat.cpp
// demonstrates relational operators
#include <iostream>
using namespace std;

int main()
{
    int numb;

    cout << "Enter a number: ";
    cin >> numb;
    cout << "numb<10 is " << (numb < 10) << endl;
    cout << "numb>10 is " << (numb > 10) << endl;
    cout << "numb==10 is " << (numb == 10) << endl;
    return 0;
}
```

İşlem Önceliği

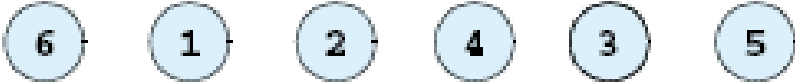
| Operators | Associativity | Type |
|------------------|---------------|----------------|
| () | left to right | parentheses |
| ++ -- | right to left | unary postfix |
| ++ -- + - (type) | right to left | unary |
| * / % | left to right | multiplicative |
| + - | left to right | additive |
| < <= > >= | left to right | relational |
| == != | left to right | equality |
| ? : | right to left | conditional |
| = += -= *= /= %= | right to left | assignment |

Fig. 4.15 Precedence and associativity of the operators discussed so far.

İşlem Önceliği

Algebra: $z = pr\%q + w/x - y$

Java: `z = p * r % q + w / x - y;`



Using redundant parentheses in complex arithmetic expressions can make the expressions clearer.

Aritmetik Atama İşleçleri

```
// assign.cpp
// demonstrates arithmetic assignment operators
#include <iostream>
using namespace std;

int main()
{
    int ans = 27;

    ans += 10;           //same as: ans = ans + 10;
    cout << ans << ", ";
    ans -= 7;            //same as: ans = ans - 7;
    cout << ans << ", ";
    ans *= 2;            //same as: ans = ans * 2;
    cout << ans << ", ";
    ans /= 3;            //same as: ans = ans / 3;
    cout << ans << ", ";
    ans %= 3;            //same as: ans = ans % 3;
    cout << ans << endl;
    return 0;
}
```

Programmers can write programs a bit faster and compilers can compile programs a bit faster when the abbreviated assignment operators are used. Some compilers generate code that runs faster when abbreviated assignment operators are used.

Artırma Azaltma İşleçleri

```
// increm.cpp
// demonstrates the increment operator
#include <iostream>
using namespace std;

int main()
{
    int count = 10;

    cout << "count=" << count << endl;    //displays 10
    cout << "count=" << ++count << endl;    //displays 11 (prefix)
    cout << "count=" << count << endl;    //displays 11
    cout << "count=" << count++ << endl;    //displays 11 (postfix)
    cout << "count=" << count << endl;    //displays 12
    return 0;
}
```

Kitaplık Fonksiyonları

```
// sqrt.cpp
// demonstrates sqrt() library function
#include <iostream>           //for cout, etc.
#include <cmath>              //for sqrt()
using namespace std;

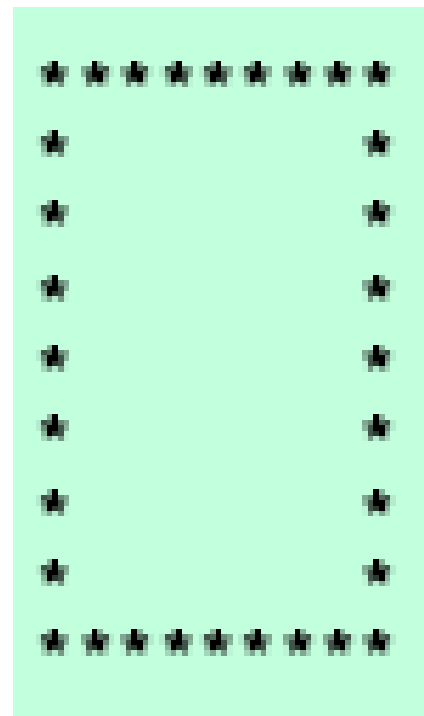
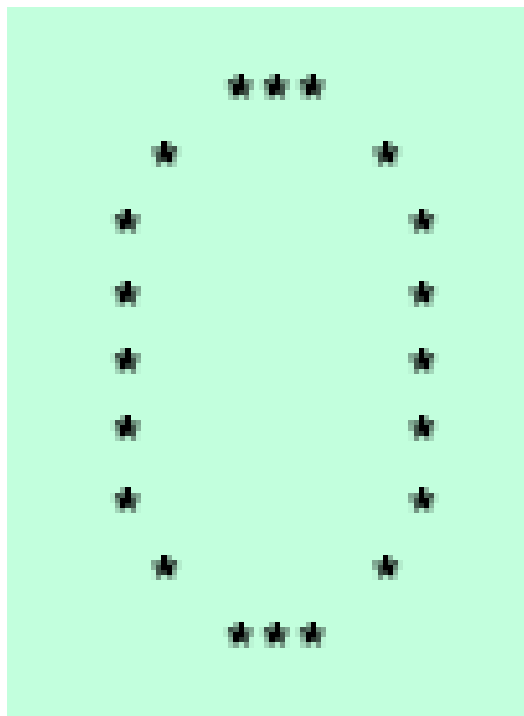
int main()
{
    double number, answer;    //sqrt() requires type double

    cout << "Enter a number: ";
    cin >> number;            //get the number
    answer = sqrt(number);     //find square root
    cout << "Square root is "
    << answer << endl;        //display it
    return 0;
}
```

| Method | Description | Example |
|---------------------|--|---|
| ceil(x) | rounds x to the smallest integer not less than x | ceil(9.2) is 10.0 ceil(-9.8) is -9.0 |
| cos(x) | trigonometric cosine of x (x in radians) | cos(0.0) is 1.0 |
| exp(x) | exponential function e^x | exp(1.0) is 2.71828 exp(2.0) is 7.38906 |
| fabs(x) | absolute value of x | fabs(5.1) is 5.1 fabs(0.0) is 0.0 fabs(-8.76) is 8.76 |
| floor(x) | rounds x to the largest integer not greater than x | floor(9.2) is 9.0 floor(-9.8) is -10.0 |
| fmod(x, y) | remainder of x/y as a floating-point number | fmod(13.657, 2.333) is 2.001 |
| log(x) | natural logarithm of x (base e) | log(2.718282) is 1.0 log(7.389056) is 2.0 |
| log10(x) | logarithm of x (base 10) | log10(10.0) is 1.0 log10(100.0) is 2.0 |
| pow(x, y) | x raised to power y (xy) | pow(2, 7) is 128 pow(9, .5) is 3 |
| sin(x) | trigonometric sine of x (x in radians) | sin(0.0) is 0 |
| sqrt(x) | square root of x | sqrt(900.0) is 30.0 sqrt(9.0) is 3.0 |
| tan(x) | trigonometric tangent of x (x in radians) | tan(0.0) is 0 |

Soruları

1. C ve C++ dillerinin tarihini araştırınız.
2. Kullanıcıdan istediği iki tamsayıyı okuyan ve bu değerlerin toplamını, farkını, ortalamasını, çarpımını, bölümünü ve bölümünden kalanını bulup ekrana yazdıran programı yazınız.
3. Aşağıdaki çıktıları alt alta üreten programı yazınız.



- 2.13** Given that $y = ax^3 + 7$, which of the following are correct Java statements for this equation?
- `y = a * x * x * x + 7;`
 - `y = a * x * x * (x + 7);`
 - `y = (a * x) * x * (x + 7);`
 - `y = (a * x) * x * x + 7;`
 - `y = a * (x * x * x) + 7;`
 - `y = a * x * (x * x + 7);`
- 2.14** State the order of evaluation of the operators in each of the following Java statements, and show the value of `x` after each statement is performed:
- `x = 7 + 3 * 6 / 2 - 1;`
 - `x = 2 % 2 + 2 * 2 - 2 / 2;`
 - `x = (3 * 9 * (3 + (9 * 3 / (3))));`
- 2.15** Write an application that displays the numbers 1 to 4 on the same line, with each pair of adjacent numbers separated by one space. Write the program using the following methods:
- Using one `System.out` statement.
 - Using four `System.out` statements.
- 2.16** Write an application that asks the user to enter two numbers, obtains the numbers from the user and prints the sum, product, difference and quotient (division) of the numbers. Use the techniques shown in Fig. 2.9.
- 2.17** Write an application that asks the user to enter two integers, obtains the numbers from the user and displays the larger number followed by the words “**is larger**” in an information message dialog. If the numbers are equal, print the message “**These numbers are equal.**” Use the techniques shown in Fig. 2.20.
- 2.18** Write an application that inputs three integers from the user and displays the sum, average, product, smallest and largest of the numbers in an information message dialog. Use the GUI techniques shown in Fig. 2.20. [Note: The calculation of the average in this exercise should result in an integer representation of the average. So, if the sum of the values is 7, the average should be 2, not 2.3333....]
- 2.19** Write an application that inputs from the user the radius of a circle and prints the circle’s diameter, circumference and area. Use the value 3.14159 for π . Use the GUI techniques shown in Fig. 2.9. [Note: You may also use the predefined constant `Math.PI` for the value of π . This constant is more precise than the value 3.14159. Class `Math` is defined in the `java.lang` package, so you do not need to `import` it.] Use the following formulas (r is the radius):

$$\begin{aligned} \text{diameter} &= 2r \\ \text{circumference} &= 2\pi r \\ \text{area} &= \pi r^2 \end{aligned}$$

Kaynaklar

- ✓ Horstmann, C., Budd, T., Big C++, John Wiley & Sons, Inc.
- ✓ Deitel, C++ How To Program, Prentice Hall
- ✓ Robert Lafore, Object Oriented Programming in C++, Macmillan Computer Publishing