Basit C Programlama Örnekleri

Bu yazıda C programalama dilini öğrenmeye yönelik örnekler ve çözümlerini göreceksiniz. Örnekler basit seviye, orta seviye ve zor seviye olarak sıralanmıştır.

**C Örnekleri:**

**C Programlama dili ile ilk programımız ekranda “Hello World” “Merhaba Dünya” örneği.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | #include <stdio.h>  int main() {     // printf() displays the string inside quotation     printf("Hello, World!");     return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3 | Hello World |

**Bu örnekte, kullanıcı tarafından girilen tamsayı bir değişkende saklanır ve ekrana yazdırılır.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #include <stdio.h>  int main() {      int number;        printf("Sayıyı Gir : ");        // reads and stores input      scanf("%d", &number);        // displays output      printf("Girdiğiniz Sayı : %d", number);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayıyı Gir : 20  Girdiğiniz Sayı : 20 |

**Bu örnekte, kullanıcı tarafından girilen iki ondalıklı sayının çarpımı hesaplanır ve ekrana yazdırılır.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #include <stdio.h>  int main() {      double a, b, sonuc;      printf("Sayı 1: ");      scanf("%lf", &a);      printf("Sayı 2: ");      scanf("%lf", &b);        sonuc = a \* b;        printf("Sonuç = %.2lf", sonuc);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Sayı 1: 3.5  Sayı 2: 2  Sonuç = 7.00 |

**Bu örnekte, bir karakterin ASCII değerini nasıl bulacağınızı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | #include <stdio.h>  int main() {      char c;      printf("Bir karakter girin : ");      scanf("%c", &c);        printf("%c karakteri ASCII değeri = %d", c, c);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Bir karakter girin : A  A karakteri ASCII değeri = 65 |

**Bu örnekte, bir tamsayı başka bir tam sayıya bölündüğünde bölümü ve kalanı bulmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | #include <stdio.h>  int main() {      int bolunen, bolen, bolum, kalan;      printf("Bölünen: ");      scanf("%d", &bolunen);      printf("Bölen: ");      scanf("%d", &bolen);        // Computes quotient      bolum = bolunen / bolen;        // Computes remainder      kalan = bolunen % bolen;        printf("Bölüm = %d\n", bolum);      printf("Kalan = %d", kalan);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Bölünen: 21  Bölen: 4  Bölüm = 5  Kalan = 1 |

**Bu örnekte, sizeof operatörünü kullanarak her bir değişkenin boyutunu değerlendirmeyi öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | #include<stdio.h>  int main() {      int intType;      float floatType;      double doubleType;      char charType;        // sizeof evaluates the size of a variable      printf("int Boyutu : %zu bytes\n", sizeof(intType));      printf("float Boyutu : %zu bytes\n", sizeof(floatType));      printf("double Boyutu : %zu bytes\n", sizeof(doubleType));      printf("char Boyutu : %zu byte\n", sizeof(charType));        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | int Boyutu : 4 bytes  float Boyutu : 4 bytes  double Boyutu : 8 bytes  char Boyutu : 1 byte |

**Bu örnekte, iki farklı teknik kullanarak C programlamasında iki sayıyı değiştirmeyi öğreneceksiniz.**

İlk  yöntemde üçüncü bir değişken kullanılarak yer değiştirme işlemi gerçekleştirirken 2. yöntemde üçüncü değişken kullanılmamaktadır.

**C Kodları: Yöntem-1 (3. Değişken Kullanarak Yer Değiştirme)**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | #include<stdio.h>  int main() {        double ilkSayi, ikinciSayi, temp;        printf("Sayı 1: ");        scanf("%lf", &ilkSayi);        printf("Sayı 2: ");        scanf("%lf", &ikinciSayi);          temp = ilkSayi;          ilkSayi = ikinciSayi;          ikinciSayi = temp;          printf("\nDeğişiklikten sonra ilkSayi = %.2lf\n", ilkSayi);        printf("Değişiklikten sonra ikinciSayi = %.2lf", ikinciSayi);        return 0;  } |

**C Kodları: Yöntem-2 (3.Değişken Kullanmadan Yer Değiştirme)**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | #include <stdio.h>  int main() {      double a, b;      printf("Sayı 1: ");      scanf("%lf", &a);      printf("Sayı 2: ");      scanf("%lf", &b);        a = a - b;        b = a + b;        a = b - a;        printf("Değişiklikten sonra, a = %.2lf\n", a);      printf("Değişiklikten sonra, b = %.2lf", b);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Sayı 1: 5  Sayı 2: 10    Değişiklikten sonra ilkSayi = 10.00  Değişiklikten sonra ikinciSayi = 5.00 |

**Bu örnekte, kullanıcı tarafından girilen bir sayının çift mi yoksa tek mi olduğunu kontrol etmeyi öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #include <stdio.h>  int main() {      int num;      printf("Sayı Gir: ");      scanf("%d", &num);        // True if num is perfectly divisible by 2      if(num % 2 == 0)          printf("%d Çift.", num);      else          printf("%d Tek.", num);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı Gir: 14  14 Çift. |

**Bu örnekte, kullanıcı tarafından girilen bir alfabenin sesli mi yoksa ünsüz mü olduğunu kontrol etmeyi öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | #include <stdio.h>  int main() {      char c;      int sesliKucukHarf, sesliBuyukHarf;      printf("Bir karakter girin: ");      scanf("%c", &c);        sesliKucukHarf = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');        sesliBuyukHarf = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');        if (sesliKucukHarf || sesliBuyukHarf)          printf("%c ünlü harf.", c);      else          printf("%c ünsüz harf.", c);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Bir Karakter Girin: E  E ünlü harf. |

**Bu örnekte, kullanıcı tarafından girilen üç sayı arasında en büyük sayıyı bulmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | #include <stdio.h>  int main() {      double n1, n2, n3;      printf("Sayı1: ");      scanf("%lf", &n1);      printf("Sayı2: ");      scanf("%lf", &n2);      printf("Sayı3: ");      scanf("%lf", &n3);        if (n1 >= n2 && n1 >= n3)          printf("Girilen En Büyük Sayı : %.2lf.", n1);        else if (n2 >= n1 && n2 >= n3)          printf("Girilen En Büyük Sayı : %.2lf.", n2);        else          printf("Girilen En Büyük Sayı : %.2lf.", n3);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Sayı1: 8  Sayı2: 15  Sayı3: 4  Girilen En Büyük Sayı : 15.00 |

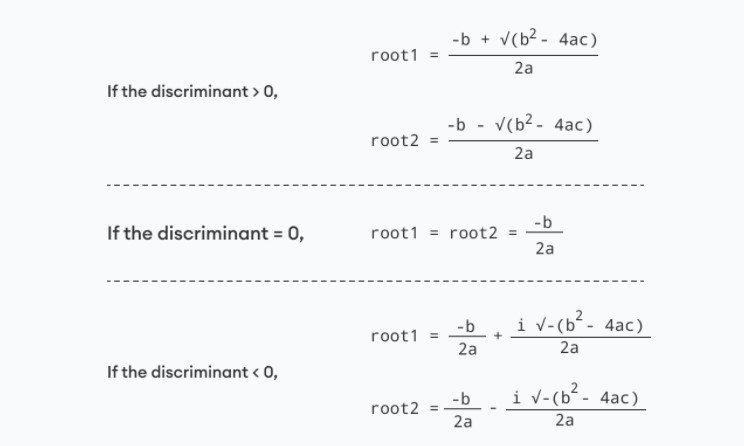
**Bu örnekte, C programlamasında ikinci dereceden bir denklemin köklerini bulmayı öğreneceksiniz.**

İkinci dereceden bir denklemin standart biçimi şöyledir:

|  |  |
| --- | --- |
| 1  2  3  4  5 | ax<sup>2</sup> + bx + c = 0, burada  a, b ve c gerçek sayılardır ve  a != 0 |

**b2-4ac** terimi, ikinci dereceden bir denklemin ayırt edicisi olarak bilinir. Köklerin doğasını anlatır.

Ayırıcı 0’dan büyükse, kökler gerçek ve farklıdır.  
Ayırıcı 0’a eşitse, kökler gerçek ve eşittir.  
Ayırıcı 0’dan küçükse, kökler karmaşık ve farklıdır.



**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37 | #include <math.h>  #include <stdio.h>  int main() {      double a, b, c, discriminant, kok1, kok2, realPart, imagPart;      printf("a: ");      scanf("%lf", &a);      printf("b: ");      scanf("%lf", &b);      printf("c: ");      scanf("%lf", &c);        discriminant = b \* b - 4 \* a \* c;        // condition for real and different roots      if (discriminant > 0) {          kok1 = (-b + sqrt(discriminant)) / (2 \* a);          kok2 = (-b - sqrt(discriminant)) / (2 \* a);          printf("root1 = %.2lf and root2 = %.2lf", kok1, kok2);      }        // condition for real and equal roots      else if (discriminant == 0) {          kok1 = kok2 = -b / (2 \* a);          printf("root1 = root2 = %.2lf;", kok1);      }        // if roots are not real      else {          realPart = -b / (2 \* a);          imagPart = sqrt(-discriminant) / (2 \* a);          printf("kök1 = %.2lf+%.2lfi ve kök2 = %.2f-%.2fi", realPart, imagPart, realPart, imagPart);      }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | a: 2  b: 4  c: 3  kök1 = -1.00+0.71i ve kök2 = -1.00-0.71i |

**Bu örnekte, kullanıcı tarafından girilen yılın artık yıl olup olmadığını kontrol etmeyi öğreneceksiniz.**

Artık yıl, yüzyıl yılları dışında (00 ile biten yıllar) 4’e bölünebilir. Yüzyıl, ancak 400 ile mükemmel bir şekilde bölünebilirse artık bir yıldır.

Örneğin,

1999 artık bir yıl değil  
2000 artık bir yıldır  
2004 artık bir yıldır

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | #include <stdio.h>  int main() {     int yil;     printf("Yıl Girin: ");     scanf("%d", &yil);         if (yil % 400 == 0) {        printf("%d is artık yıl.", yil);     }       else if (yil % 100 == 0) {        printf("%d artık yıl değil.", yil);     }       else if (yil % 4 == 0) {        printf("%d artık yıl.", yil);     }       else {        printf("%d artık yıl değil.", yil);     }       return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Yıl Girin: 2021  2021 artık yıl değil. |

**Bu örnekte, bir sayının (kullanıcı tarafından girilen) negatif mi yoksa pozitif mi olduğunu kontrol etmeyi öğreneceksiniz.**

Bu program kullanıcıdan bir sayı alır ve bu sayının pozitif mi, negatif mi yoksa sıfır mı olduğunu kontrol eder.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #include <stdio.h>  int main() {      double num;      printf("Sayı: ");      scanf("%lf", &num);      if (num <= 0.0) {          if (num == 0.0)              printf("0 girdiniz..");          else              printf("Negatif sayı girdiniz..");      } else          printf("Pozitif sayı girdiniz..");      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı: -14  Negatif sayı girdiniz. |

**Bu örnekte, kullanıcı tarafından girilen bir karakterin bir alfabe olup olmadığını kontrol etmeyi öğreneceksiniz.**

C programlamasında, bir karakter değişkeni, o karakterin kendisinden ziyade bir ASCII değerini (0 ile 127 arasında bir tam sayı) tutar.

Küçük harfli alfabenin ASCII değeri 97’den 122’ye kadardır. Ve büyük harfli alfabenin ASCII değeri 65’den 90’a kadardır.

Kullanıcı tarafından girilen karakterin ASCII değeri 97 ila 122 veya 65 ila 90 aralığındaysa, bu sayı bir alfabedir.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | #include <stdio.h>  int main() {      char c;      printf("Bir karakter girin: ");      scanf("%c", &c);        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))          printf("%c alfabe karakteri.", c);      else          printf("%c alfabede değil.", c);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | Bir karakter girin: ;  ; alfabede değil.    Bir karakter girin: A  A alfabe karakteri. |

**Bu örnekte, kullanıcı tarafından girilen doğal sayıya kadar sayıların toplamını hesaplamayı öğreneceksiniz.**

Pozitif sayılar 1, 2, 3 … doğal sayılar olarak bilinir. Örnek olarak kullanıcı 10 sayısı girdiyse;

10’a kadar olan doğal sayıların toplamı:

|  |  |
| --- | --- |
| 1  2  3 | sum = 1 + 2 + 3 + ... + 10 |

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #include <stdio.h>  int main() {      int n, i, toplam = 0;        printf("Sayı Gir: ");      scanf("%d", &n);        for (i = 1; i <= n; ++i) {          toplam += i;      }        printf("Toplam = %d", toplam);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı Gir: 4  Toplam = 10 |

**Bu örnekte, kullanıcı tarafından girilen bir sayının faktöriyelini hesaplamayı öğreneceksiniz.**

Pozitif bir n sayısının faktöriyeli şu şekilde verilir:

|  |  |
| --- | --- |
| 1  2  3 | <span class="hljs-function"><span class="hljs-params">(n!)</span> </span>= <span class="hljs-number">1</span> \* <span class="hljs-number">2</span> \* <span class="hljs-number">3</span> \* <span class="hljs-number">4.</span>...n |

Negatif bir sayının faktöriyeli yoktur. Ve 0’ın faktöriyeli 1’dir.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | #include <stdio.h>  int main() {      int n, i;      unsigned long long fact = 1;      printf("Sayı : ");      scanf("%d", &n);        // shows error if the user enters a negative integer      if (n < 0)          printf("Hata! Negatif sayılarda faktoriyel hesaplanamaz.");      else {          for (i = 1; i <= n; ++i) {              fact \*= i;          }          printf("%d sayısının faktoriyeli  = %llu", n, fact);      }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı : 5  5 sayısının faktoriyeli = 12 |

**Bu örnekte, kullanıcı tarafından girilen bir sayının çarpım tablosunu oluşturmayı öğreneceksiniz.**

Aşağıdaki program kullanıcıdan bir tamsayı girdisi alır ve 10’a kadar çarpım tablosunu oluşturur.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | #include <stdio.h>  int main() {      int n, i;      printf("Sayı Gir: ");      scanf("%d", &n);      for (i = 1; i <= 10; ++i) {          printf("%d \* %d = %d \n", n, i, n \* i);      }      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | Sayı Gir: 5  5 \* 1 = 5  5 \* 2 = 10  5 \* 3 = 15  5 \* 4 = 20  5 \* 5 = 25  5 \* 6 = 30  5 \* 7 = 35  5 \* 8 = 40  5 \* 9 = 45  5 \* 10 = 50 |

**Bu örnekte, ilk n sayının (kullanıcı tarafından girilen) Fibonacci dizisini görüntülemeyi öğreneceksiniz.**

Fibonacci dizisi, sonraki terimin önceki iki terimin toplamı olduğu bir dizidir. Fibonacci dizisinin ilk iki terimi 0 ve ardından 1’dir.

|  |  |
| --- | --- |
| 1  2  3 | Fibonacci dizisi: 0, 1, 1, 2, 3, 5, 8, 13, 21 |

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | #include <stdio.h>  int main() {      int i, n, t1 = 0, t2 = 1, nextTerm;      printf("Terim sayısını girin: ");      scanf("%d", &n);      printf("Fibonacci Series: ");        for (i = 1; i <= n; ++i) {          printf("%d, ", t1);          nextTerm = t1 + t2;          t1 = t2;          t2 = nextTerm;      }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Terim sayısını girin: 5  Fibonacci Series: 0, 1, 1, 2, 3, |

Döngüler ve karar verme ifadeleri kullanarak iki tam sayının (hem pozitif hem de negatif tam sayılar için) OBEB hesaplamasının farklı yollarına ilişkin örnekler.

C programlamada en büyük ortak böleni bulmanın birçok yolu vardır.

**Örnek1:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | #include <stdio.h>  int main()  {      int n1, n2, i, gcd;        printf("1. Sayı: ");      scanf("%d", &n1);      printf("2. Sayı: ");      scanf("%d", &n2);        for(i=1; i <= n1 && i <= n2; ++i)      {            if(n1%i==0 && n2%i==0)              gcd = i;      }        printf("%d ve  %d sayılarının OBEB' i = %d", n1, n2, gcd);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | 1. Sayı: 20  2. Sayı: 30  20 ve 30 sayılarının OBEB' i = 10 |

Bu programda, kullanıcı tarafından girilen iki tamsayı n1 ve n2 değişkenlerinde saklanır ve daha sonra, i n1 ve n2’den küçük olana kadar for döngüsü yinelenir.

Her yinelemede, hem n1 hem de n2 tam olarak i ile bölünebiliyorsa, i’nin değeri gcd’ye atanır.

For döngüsü tamamlandığında, iki sayının en büyük ortak böleni değişken gcd’de saklanır.

While Döngüsü ile OBEB Hesaplama

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | #include <stdio.h>  int main()  {      int n1, n2;        printf("İki sayı girin : ");      scanf("%d %d",&n1,&n2);        while(n1!=n2)      {          if(n1 > n2)              n1 -= n2;          else              n2 -= n1;      }      printf("OBEB= %d",n1);        return 0;  } |

**Bu örnekte, İngilizce alfabenin tüm harflerini yazdırmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | #include <stdio.h>  int main() {      char c;      for (c = 'A'; c <= 'Z'; ++c)          printf("%c ", c);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3 | <samp>A B C D E F G H I J K L M N O P Q R S T U V W X Y Z</samp> |

Bu programda, İngilizce alfabeyi büyük harfle görüntülemek için for döngüsü kullanılır.

İşte yukarıdaki programın küçük bir değişikliği. Program, kullanıcı tarafından verilen girişe bağlı olarak İngilizce alfabeyi büyük veya küçük harf olarak görüntüler.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | #include <stdio.h>  int main() {      char c;      printf("Küçük harfler için (k) harfine basın.\n");      printf("Büyük harfler için (b) harfine basın.. \n");      scanf("%c", &c);        if (c == 'b' || c == 'B') {          for (c = 'A'; c <= 'Z'; ++c)              printf("%c ", c);      } else if (c == 'k' || c == 'K') {          for (c = 'a'; c <= 'z'; ++c)              printf("%c ", c);      } else {          printf("Hatalı karakter girdiniz.");      }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Küçük harfler için (k) harfine basın.  Büyük harfler için (b) harfine basın..  k  a b c d e f g h i j k l m n o p q r s t u v w x y z |

**Bu örnekte, kullanıcı tarafından girilen bir tam sayıdaki basamak sayısını saymayı öğreneceksiniz.**

Bu program kullanıcıdan bir tamsayı alır ve basamak sayısını hesaplar. Örneğin: Kullanıcı 3452 girerse, programın çıktısı 4 olacaktır.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | #include <stdio.h>  int main() {      long long n;      int count = 0;      printf("Sayı: ");      scanf("%lld", &n);          while (n != 0) {          n /= 10;     // n = n/10          ++count;      }        printf("Basamak Sayısı: %d", count);  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı: 3452  Basamak Sayısı: 4 |

Kullanıcı tarafından girilen tam sayı, n değişkeninde saklanır. Daha sonra while döngüsü, test ifadesi n! = 0, 0 olarak değerlendirilir (yanlış).

İlk yinelemeden sonra, n’nin değeri 345 olacak ve sayı 1’e yükseltilecek.  
İkinci yinelemeden sonra, n’nin değeri 34 olur ve sayı 2’ye çıkarılır.  
Üçüncü yinelemeden sonra, n’nin değeri 3 olacak ve sayı 3’e yükseltilecek.  
Dördüncü yinelemeden sonra, n’nin değeri 0 olur ve sayı 4’e çıkarılır.  
Daha sonra döngünün test ifadesi yanlış olarak değerlendirilir ve döngü sona erer.  
Bu örnekte, kullanıcı tarafından girilen sayıyı tersine çevirmeyi öğreneceksiniz.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | #include <stdio.h>  int main() {      int n, ters = 0, kalan;      printf("Sayı: ");      scanf("%d", &n);      while (n != 0) {          kalan = n % 10;          ters = ters \* 10 + kalan;          n /= 10;      }      printf("Ters = %d", ters);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı: 2021  Ters = 1202 |

Bu program kullanıcıdan bir tamsayı girdisi alır. Daha sonra, **n! = 0** (n eşit değil 0) **false** olana kadar **while** döngüsü kullanılır.

Döngünün her yinelemesinde, n 10’a bölündüğünde kalan hesaplanır ve n’nin değeri 10 kat azaltılır.

Döngünün içinde, tersine çevrilen sayı şu şekilde hesaplanır:

|  |  |
| --- | --- |
| 1  2  3 | ters= ters\*<span class="hljs-number">10</span> + kalan; |

**Bu örnekte C Programlama dilinde pow() fonksiyonu kullanmadan üs alma işlemini göreceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | #include <stdio.h>  int main() {      int base, exp;      long long sonuc = 1;      printf("Sayıyı Gir: ");      scanf("%d", &base);      printf("Üs Değeri Gir: ");      scanf("%d", &exp);        while (exp != 0) {          sonuc \*= base;          --exp;      }      printf("Sonuç = %lld", sonuc);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Sayıyı Gir: 7  Üs Değeri Gir: 2  Sonuç = 49 |

**Bu örnekte, kullanıcı tarafından girilen sayının bir palindrom sayı olup olmadığını kontrol etmeyi öğreneceksiniz.**

**Palindromik sayı**, iki taraftan okunduğu zaman okunuş yönüyle aynı olan sayılardır.

**Örnek**: 1, 4, 8, 99, 101, 363, 4004, 9889, 13531 vb.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | #include <stdio.h>  int main() {      int n, reversedN = 0, remainder, originalN;      printf("Sayı: ");      scanf("%d", &n);      originalN = n;        while (n != 0) {          remainder = n % 10;          reversedN = reversedN \* 10 + remainder;          n /= 10;      }        if (originalN == reversedN)          printf("%d sayısı palindromik sayıdır.", originalN);      else          printf("%d sayısı palindromik sayı değildir.", originalN);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı: 12321  12321 sayısı palindromik sayıdır. |

**Bu örnekte, kullanıcı tarafından girilen bir tamsayının asal sayı olup olmadığını kontrol etmeyi öğreneceksiniz.**

Bir asal sayı, yalnızca 1’e ve kendisine bölünebilen pozitif bir tam sayıdır. Örneğin: 2, 3, 5, 7, 11, 13, 17

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | #include <stdio.h>  int main() {      int n, i, flag = 0;      printf("Pozitif bir sayı girin: ");      scanf("%d", &n);        for (i = 2; i <= n / 2; ++i) {            if (n % i == 0) {              flag = 1;              break;          }      }        if (n == 1) {          printf("1 ne asal ne de bileşiktir.");      }      else {          if (flag == 0)              printf("%d sayısı Asal Sayıdır.", n);          else              printf("%d sayısı Asal Sayı değildir.", n);      }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Pozitif bir sayı girin: 29  29 sayısı Asal Sayıdır. |

**Bu örnekte, kullanıcı tarafından girilen iki sayı arasındaki tüm asal sayıları yazdırmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36 | #include <stdio.h>    int main() {     int low, high, i, flag;     printf("Sayı 1: ");     scanf("%d", &low);     printf("Sayı 2: ");     scanf("%d", &high);     printf(" %d ve  %d arasındaki asal sayılar: ", low, high);       while (low < high) {        flag = 0;          if (low <= 1) {           ++low;           continue;        }          for (i = 2; i <= low / 2; ++i) {             if (low % i == 0) {              flag = 1;              break;           }        }          if (flag == 0)           printf("%d ", low);          ++low;     }       return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Sayı 1: 5  Sayı 2: 20  5 ve 20 arasındaki asal sayılar: 5 7 11 13 17 19 |

**Bu örnekte, kullanıcı tarafından girilen bir tamsayının Armstrong sayı olup olmadığını kontrol etmeyi öğreneceksiniz.**

İlk örnek 3 basamaklı bir sayının Armstrong sayı olup olmadığını kontrol ederken, ikinci örneğimizde n basamaklı yani kullanıcının girdiği herhangi bir sayıya göre işlem yapmaktadır.

Pozitif bir tamsayı, Armstrong numarası olarak adlandırılır (n sırasına göre)

|  |  |
| --- | --- |
| 1  2  3 | abcd... = a<sup>n</sup> + b<sup>n</sup> + c<sup>n</sup> + d<sup>n</sup> + |

Armstrong sayısının 3 basamaklı olması durumunda, her basamağın küplerinin toplamı, sayının kendisine eşittir. Örneğin, 153 bir Armstrong numarasıdır çünkü

|  |  |
| --- | --- |
| 1  2  3 | 153 = 1\*1\*1 + 5\*5\*5 + 3\*3\*3 |

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | #include <stdio.h>  int main() {      int num, originalNum, remainder, result = 0;      printf("3 Basamaklı sayı girin: ");      scanf("%d", &num);      originalNum = num;        while (originalNum != 0) {          remainder = originalNum % 10;           result += remainder \* remainder \* remainder;           originalNum /= 10;      }        if (result == num)          printf("%d Armstrong sayıdır.", num);      else          printf("%d Armstrong sayı değildir.", num);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | 3 Basamaklı sayı girin: 371  371 Armstrong sayıdır. |

Armstrong sayısını kontrol edin (n) basamak.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | #include <math.h>  #include <stdio.h>    int main() {     int num, originalNum, remainder, n = 0;     float result = 0.0;       printf("Sayı girin: ");     scanf("%d", &num);       originalNum = num;       for (originalNum = num; originalNum != 0; ++n) {         originalNum /= 10;     }       for (originalNum = num; originalNum != 0; originalNum /= 10) {         remainder = originalNum % 10;          result += pow(remainder, n);     }       if ((int)result == num)      printf("%d sayısı Armstrong sayıdır.", num);     else      printf("%d sayısı Armstrong sayı değildir.", num);     return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı girin: 1634  1634 sayısı Armstrong sayıdır. |

Bu programda, önce bir tamsayının hane sayısı hesaplanır ve n’de saklanır. Ve pow () işlevi, ikinci for döngüsünün her yinelemesinde tek tek basamakların gücünü hesaplamak için kullanılır.  
Bu örnekte kullanıcının seçeceği işlem türüne göre ( + , – , \* , / ) ve kullanıcının girdiği sayılara göre aritmetiksel işlemi gerçekleştiren C programını öğreneceksiniz.

Kullanıcı yanlış operatör seçimi yaptığında uyarılacaktır.

Programda **switch-case** yapısı kullanılmıştır.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | #include <stdio.h>  int main() {      char operator;      double sayi1, sayi2;      printf("Yapmak istediğiniz işlem (+, -, \*, /): ");      scanf("%c", &operator);      printf("Sayı1: ");      scanf("%lf", &sayi1);      printf("Sayı2: ");      scanf("%lf", &sayi2);        switch (operator) {      case '+':          printf("%.1lf + %.1lf = %.1lf", sayi1, sayi2, sayi1 + sayi2);          break;      case '-':          printf("%.1lf - %.1lf = %.1lf", sayi1, sayi2, sayi1 - sayi2);          break;      case '\*':          printf("%.1lf \* %.1lf = %.1lf", sayi1, sayi2, sayi1 \* sayi2);          break;      case '/':          printf("%.1lf / %.1lf = %.1lf", sayi1, sayi2, sayi1 / sayi2);          break;      default:          printf("Yanlış operatör seçimi!!!");      }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Yapmak istediğiniz işlem (+, -, \*, /): \*  Sayı1: 12  Sayı2: 10  12.0 \* 10.0 = 120.0 |

Daha önceki örneklerde kullanıcının girdiği iki sayı arasındaki asal sayıları listeleyen C programını oluşturmuştuk.

Bu örnekte aynı işlemi **Fonksiyon** kullanarak nasıl yapabileceğini öğreneceksiniz.

**C kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | #include <stdio.h>  int checkPrimeNumber(int n);  int main() {      int n1, n2, i, flag;      printf("Sayı 1: ");      scanf("%d", &n1);      printf("Sayı 2: ");      scanf("%d",&n2);      printf("%d ve  %d arasındaki Asal Sayılar: ", n1, n2);      for (i = n1 + 1; i < n2; ++i) {            flag = checkPrimeNumber(i);            if (flag == 1)              printf("%d ", i);      }      return 0;  }    int checkPrimeNumber(int n) {      int j, flag = 1;      for (j = 2; j <= n / 2; ++j) {          if (n % j == 0) {              flag = 0;              break;          }      }      return flag;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | Sayı 1: 50  Sayı 2: 100  50 ve 100 arasındaki Asal Sayılar: 53 59 61 67 71 73 79 83 89 97 |

**Bu örnekte, özyinelemeli(recursive) bir fonksiyon kullanarak doğal sayıların toplamını bulmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | #include <stdio.h>  int sayiTopla(int n);  int main() {      int num;      printf("Sayı Gir: ");      scanf("%d", &num);      printf("Toplam = %d", sayiTopla(num));      return 0;  }    int sayiTopla(int n) {      if (n != 0)          return n + sayiTopla(n - 1);      else          return n;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Sayı Gir: 20  Toplam = 210 |

Başlangıçta sayiTopla() main () ‘den çağrılır ve 20 argüman olarak iletilir.

20 sayısı sayiTopla(19) sonucuna eklenir.

sayiTopla() ‘den sayiTopla()’ ye bir sonraki fonksiyon çağrısında, sayiTopla(18) sonucuna eklenen 19 geçilir. Bu süreç n, 0’a eşit olana kadar devam eder.

N 0’a eşit olduğunda, özyinelemeli çağrı yoktur. Bu, sonuçta tam sayıların toplamını main () işlevine döndürür.

Bu örnekte, özyineleme kullanarak kullanıcı tarafından girilen negatif olmayan bir tamsayının faktöriyelini bulmayı öğreneceksiniz.

Pozitif bir n sayısının faktöriyeli şu şekilde verilir:

|  |  |
| --- | --- |
| 1  2  3  4 | <span class="hljs-function"><span class="hljs-params">(n!)</span> </span>= <span class="hljs-number">1</span> \* <span class="hljs-number">2</span> \* <span class="hljs-number">3</span> \* <span class="hljs-number">4</span> \*...  \* n |

Negatif bir sayının faktöriyeli yoktur. Ve 0’ın faktöriyeli 1’dir.

**Bu örnekte özyinelemeyi kullanarak bir sayının faktöriyelini bulmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | #include<stdio.h>  long int factor(int n);  int main() {      int n;      printf("Pozitif bir sayı girin: ");      scanf("%d",&n);      printf(" %d! = %ld", n, factor(n));      return 0;  }    long int factor(int n) {      if (n>=1)          return n\*factor(n-1);      else          return 1;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4 | Pozitif bir sayı girin: 5  5! = 120 |

**Bu örnekte, dizileri kullanarak kullanıcı tarafından girilen n sayıda öğenin ortalamasını hesaplamayı öğreneceksiniz.**

Örnekte kullanıcıdan kaç elemanlı bir dizi oluşturulacağı bilgisi alınmaktadır.

Bu kısımda 1-100 arası bir değer girilmesi istenerek, daha büyük bir değer girilmesi durumunda değer alma işlemi tekrarlanacaktır.

Daha sonra girilen eleman adeti kadar sayı istenerek dizi ortalaması hesaplanacaktır.

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | #include <stdio.h>  int main() {      int n, i;      float num[100], sum = 0.0, avg;        printf("Dizi Eleman Sayısı Girin (1-100): ");      scanf("%d", &n);        while (n > 100 || n < 1) {          printf("Hata! Eleman Sayısı (1 ila 100) aralığında olmalıdır.\n");          printf("Tekrar Sayı Girin: ");          scanf("%d", &n);      }        for (i = 0; i < n; ++i) {          printf("%d. Sayı Gir: ", i + 1);          scanf("%f", &num[i]);          sum += num[i];      }        avg = sum / n;      printf("Ortalama = %.2f", avg);      return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | Dizi Eleman Sayısı Girin (1-100): 120  Hata! Eleman Sayısı (1 ila 100) aralığında olmalıdır.  Tekrar Sayı Girin: 5  1. Sayı Gir: 50  2. Sayı Gir: 100  3. Sayı Gir: 80  4. Sayı Gir: 60  5. Sayı Gir: 75  Ortalama = 73.00 |

**Bu C programında Kullanıcının girdiği sayılarla oluşturulan bir dizide en büyük değeri bulmayı öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | #include <stdio.h>  int main() {      int i, n;      float arr[100];      printf("Dizi Eleman Sayısını Girin: ");      scanf("%d", &n);        for (i = 0; i < n; ++i) {          printf("Sayı%d: ", i + 1);          scanf("%f", &arr[i]);      }        for (i = 1; i < n; ++i) {          if (arr[0] < arr[i])              arr[0] = arr[i];      }        printf("En Büyük Değer = %.2f", arr[0]);        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | Dizi Eleman Sayısını Girin: 5  Sayı1: 35  Sayı2: 45  Sayı3: 100  Sayı4: 20  Sayı5: 50  En Büyük Değer = 100.00 |

**Bu örnekte, iki boyutlu diziler kullanarak C programlamada iki matris eklemeyi öğreneceksiniz.**

**C Kodları:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | #include <stdio.h>  int main() {      int r, c, a[100][100], b[100][100], sum[100][100], i, j;      printf("Satır Sayısı: ");      scanf("%d", &r);      printf("Sütun Sayısı: ");      scanf("%d", &c);        printf("\n1. Matrix:\n");      for (i = 0; i < r; ++i)          for (j = 0; j < c; ++j) {              printf("a%d%d Gir: ", i + 1, j + 1);              scanf("%d", &a[i][j]);          }        printf("2. Matrix:\n");      for (i = 0; i < r; ++i)          for (j = 0; j < c; ++j) {              printf("a%d%d Gir: ", i + 1, j + 1);              scanf("%d", &b[i][j]);          }        for (i = 0; i < r; ++i)          for (j = 0; j < c; ++j) {              sum[i][j] = a[i][j] + b[i][j];          }        printf("\nToplam: \n");      for (i = 0; i < r; ++i)          for (j = 0; j < c; ++j) {              printf("%d   ", sum[i][j]);              if (j == c - 1) {                  printf("\n\n");              }          }        return 0;  } |

**Ekran Çıktısı:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | Satır Sayısı: 2  Sütun Sayısı: 3    1. Matrix:  a11 Gir: 10  a12 Gir: 4  a13 Gir: 5  a21 Gir: 6  a22 Gir: 2  a23 Gir: 4  2. Matrix:  a11 Gir: 7  a12 Gir: 8  a13 Gir: 6  a21 Gir: 2  a22 Gir: 1  a23 Gir: 4    Toplam:  17 12 11    8 3 8 |