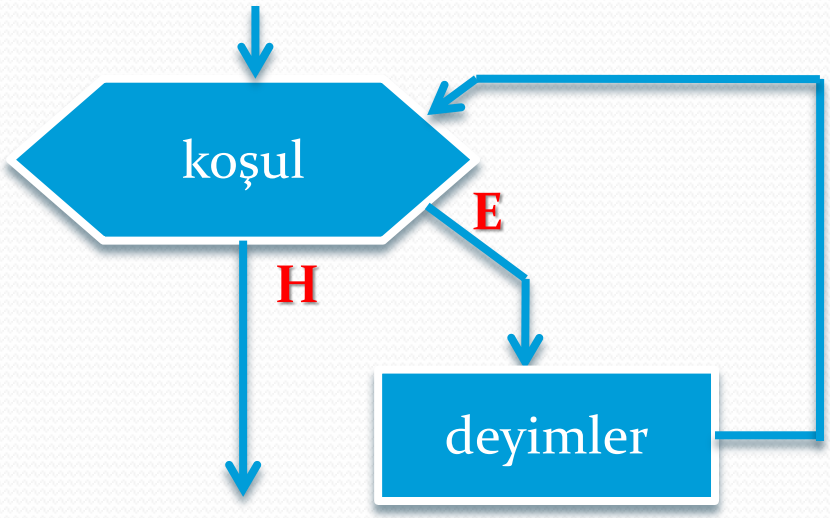
PROGRAMLAMA TEMELLERİ

Döngüler

while

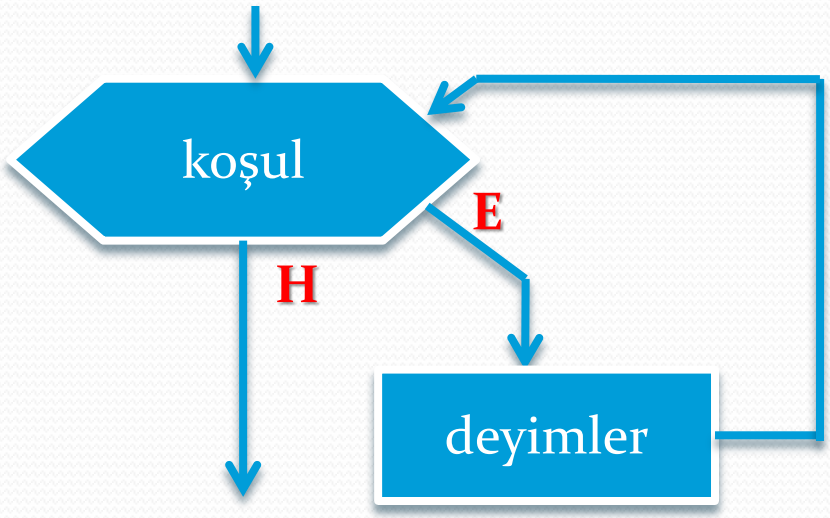
Döngüler: WHILE

- Bir koşul gerçekleştiği sürece çalıştırılması gereken komutlar varsa kullanırız.
- Koşula etki eden etmenler döngü içerisinde değişikliğe uğrayabilir.
- En esnek ve verimli döngüdür.



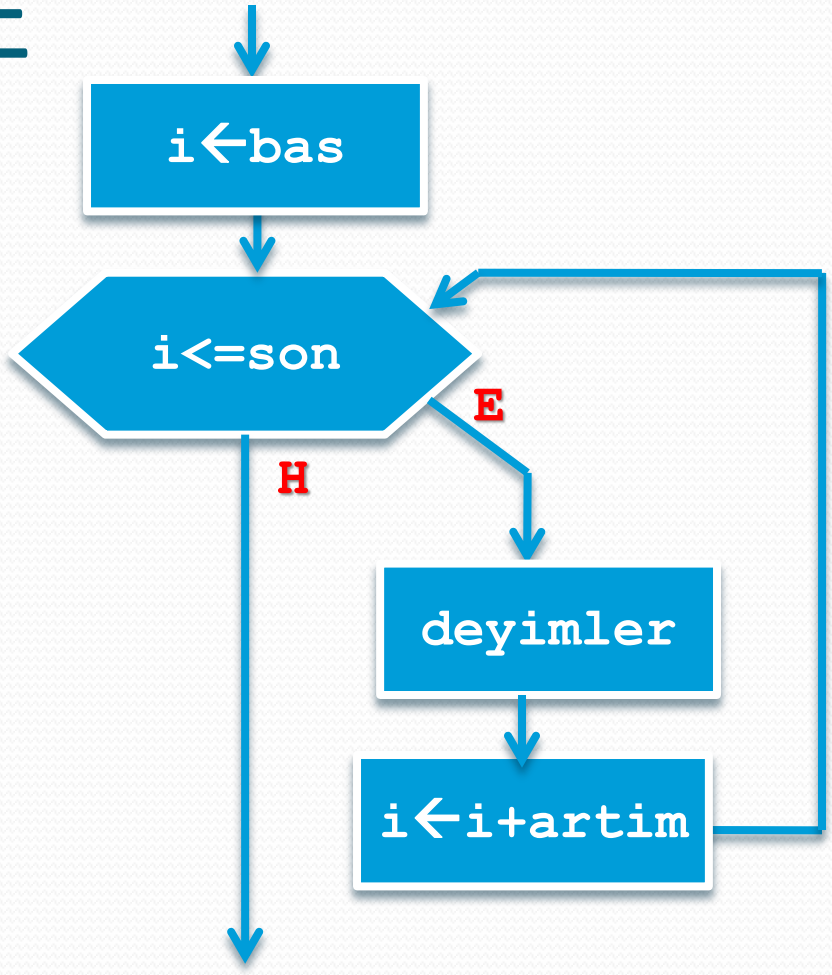
Döngüler: WHILE

- Döngüye başlamadan koşul kontrol edilir.
- Eğer koşul sağlanıyorsa ve sonrasında koşul sağlandığı yani doğru olduğu sürece döngü komutlar ı çalıştırılır.
- Eğer koşul baştan sağlanmıyorsa komutlar hiç çalışmaz.



FOR yerine WHILE

- Sabit adımlı döngü (FOR) yerine kullanılabilir.
- Bazı programlama dillerinde FOR döngüleri bu kadar esnek olmadığından yerine WHILE kullanılır.
- FOR yerine kullanılacaksa sayacın ilk değer ataması ve sayacın artırımı elle yapılır.



C++'ta while

- Bir while komutu tekrar eden deyim/leri kontrol eden bir ifade içerir.
- Yazım

```
while ( <koşul> )
{
   <deyim/ler>;
}
```

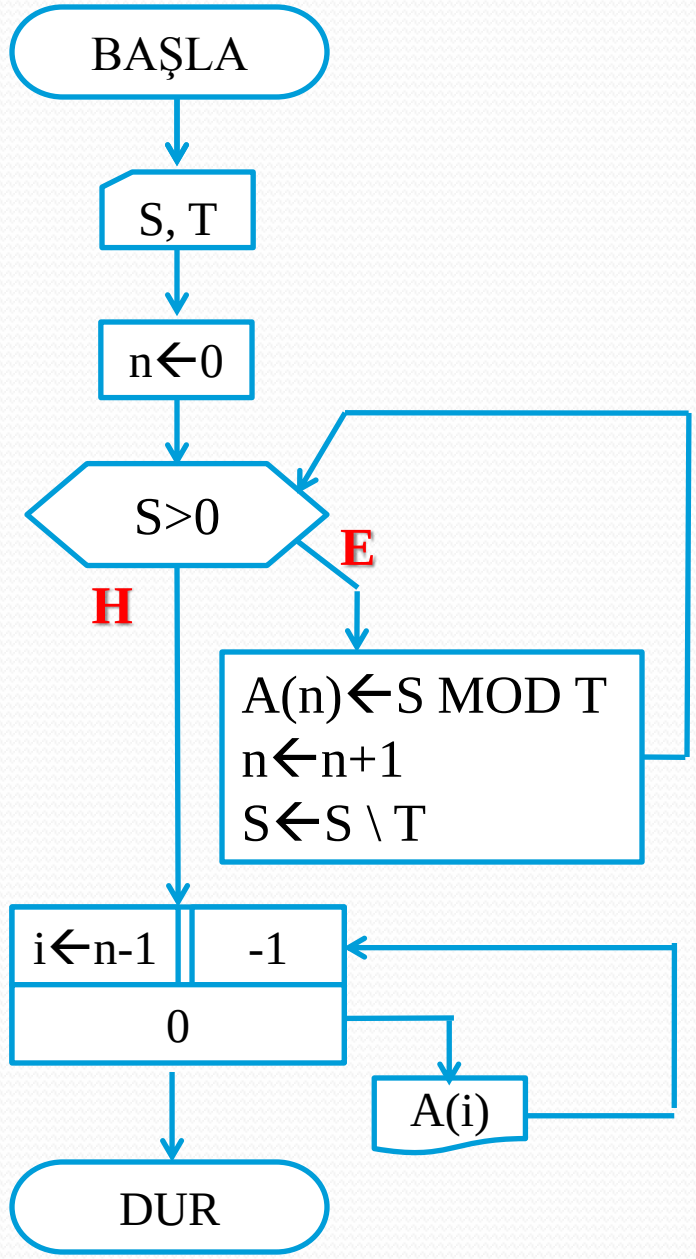
- Blok içerisindeki deyim/ler koşul sağlandığı sürece çalıştırılır.
- Koşul bu deyimlerden önce değerlendirilir.
- Eğer koşul başlangıçta sağlanmazsa deyim/ler hiç çalıştırılmaz.

Örnek:Taban Çevirme

- Klavyeden girilen onluk tabandaki bir sayıyı, yine klavyeden girilen 2 ilâ 9 arasında bir tabana çevirip sonucu ekrana yazan program.
- **Algoritma**:
- Sayı ve taban okunur.
- Sayı 0'dan büyük olduğu sürece;
  - Sayının tabana göre modu alınır ve bir dizide saklanır.
  - Sayı tabana kalansız bölünür.
- Dizi yazdırılır.

Örnek: Taban Çevirme

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int s, t, n, A[16], i;
    cout << "Sayi: "; cin >> s;
    cout << "Taban: "; cin >>t;
    n=0;
    while (s>0)
    {
      A[n]= s % t;
      n++;
      s=s/t;
    }
    for (i=n-1; i>=0; i--)
       cout << A[i];
    cout<<endl;
    system("PAUSE");
    return EXIT_SUCCESS;
}
```

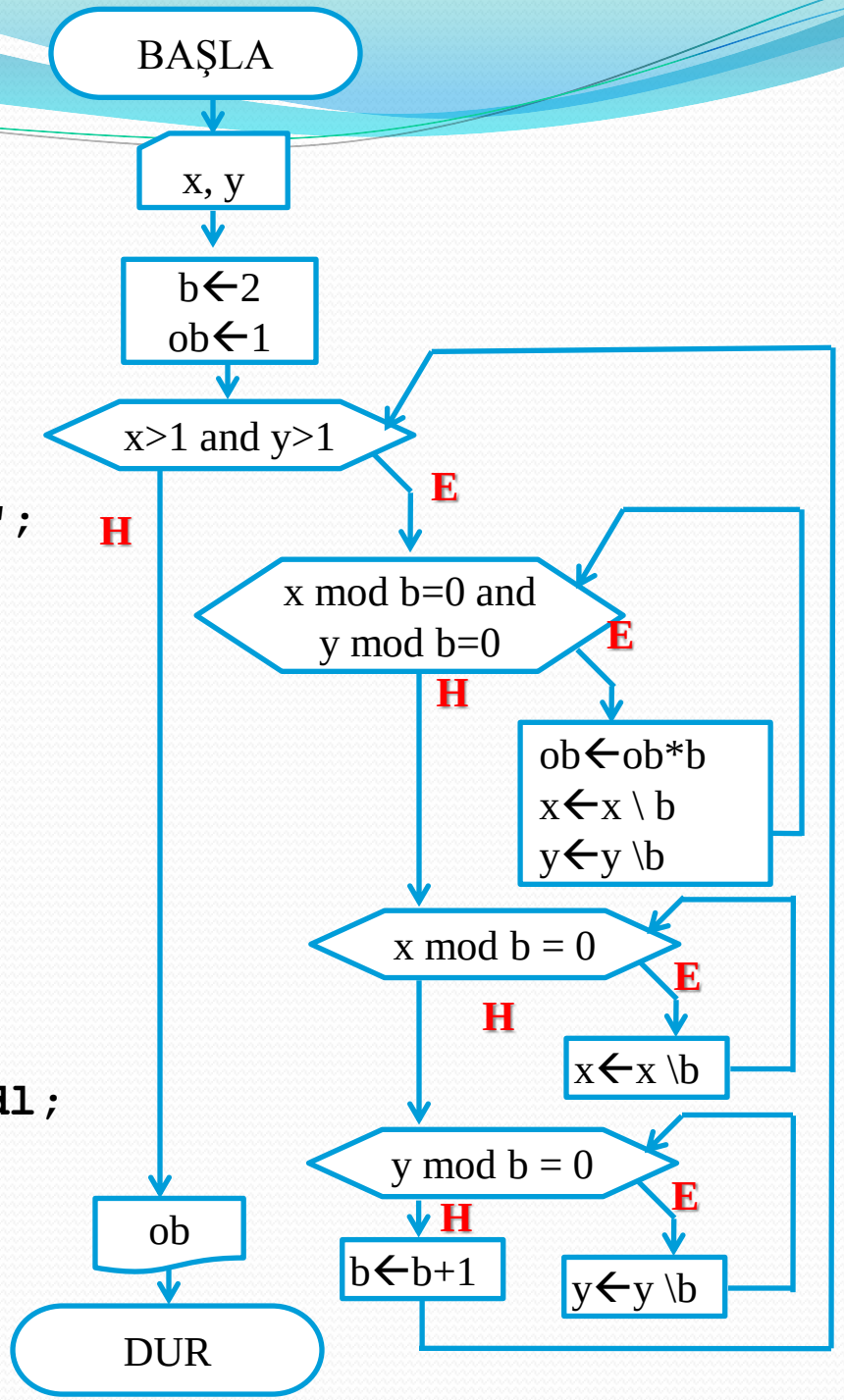


Örnek: OBEB

- Klavyeden okutulan iki pozitif tamsayının ortak bölenlerinin en büyüğünü bulup ekrana yazan program.
- **Algoritma**:
- **İki tamsayı: X ve Y okutulur.**
- **Bölen 2, OBEB 1 olarak kabul edilir.**
- **X ve Y 1'den büyük olduğu sürece;**
  - **X ve Y Bölen'e bölündüğü sürece;**
    - **OBEB'i Bölen ile çarp,**
    - **X'i Bölen'e böl.**
    - **Y'yi Bölen'e böl.**
  - **X Bölen'e bölündüğü sürece; X'i Bölen'e böl.**
  - **Y Bölen'e bölündüğü sürece; Y'yi Bölen'e böl.**
  - **Bölen'i 1 artır.**
- **OBEB yazdırılır.**

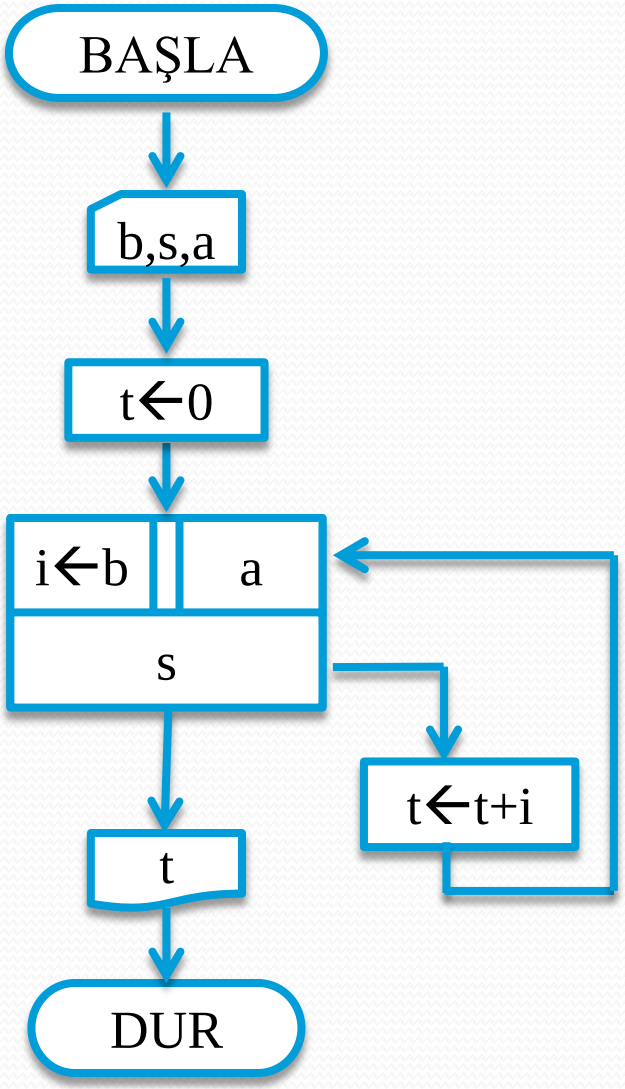
Örnek: OBEB

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    int x, y, ob=1, b=2;
    cout << "2 pozitif tamsayi: ";
    cin >> x >> y;
    while ( (x>1) && (y>1) )
    {
          while ( (x%b==0) &&
  (y%b==0))
          {
                ob*=b;
                x/=b;
                y/=b;
          }
          while (x%b==0) x/=b;
          while (y%b==0) y/=b;
          b++;
    }
    cout << "OBEB: " << ob << endl;
    system("PAUSE");
    return EXIT_SUCCESS;
}
```



Örnek: Gauss

- Normal olarak Gauss toplamları sabit adımlı döngü (for) ile gösterilir.
- Ama while ile de göstermek mümkündür.



Örnek: Gauss

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
    long t=0;
    int i, b, s, a;
    cout << "Baslangic: "; cin >> b;
    cout << "Son      : "; cin >> s;
    cout << "Artim    : "; cin >> a;
    i=b;
    while (i<=s)
    {
          t+=i;
          i+=a;
    }
    cout << "Gauss Toplami: " << t << endl;
    system("PAUSE");
    return EXIT_SUCCESS;
}
```