Şərt strukturları
(if, if-else, if-else-if, switch-case)

Real həyatda elə məqamlar olur ki, biz hər hansı qərarları vermək istəyərkən həmin qərarlardan öncə müəyyən şərtlər ortaya çıxar. Məsələn, işıqforda yaşıl işıq yansa getmək qərarını, qırmızı işıq yansa dayanmaq qərarını verə bilərik.
Oxşar hallar həmçinin proqramlaşdırmada da o vaxt ortaya çıxır ki, biz hər hansı kod bloku müəyyən şərtin doğru olub-olmaması ilə icra etməli oluruq.

if strukturu

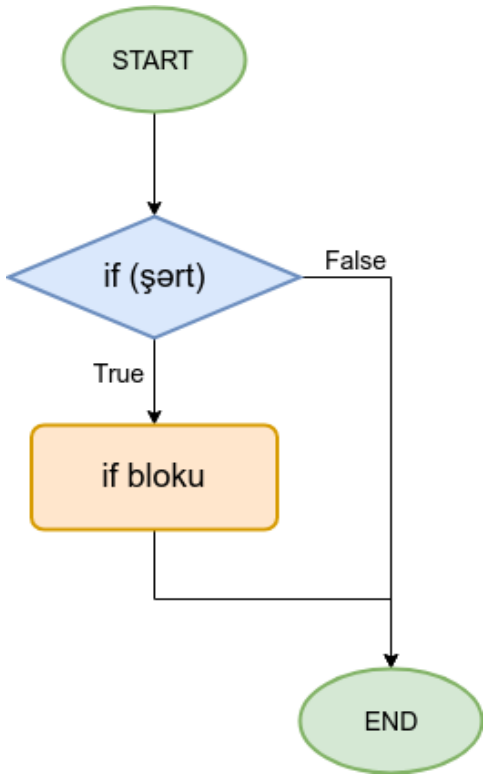
Proqramlaşdırmada ən sadə şərt strukturu **if** strukturudur. Bu struktur əsasında müəyyən şərtin doğru olması (**True**) ilə növbəti kod bloku icra olunur, əks halda (**False**) icra olunmur.

Sintaksı (quruluşu).

```
if(şərt)
{
      // if bloku
      // bu hissədəki kodlar o vaxt icra olunur ki,
      // if mötərizəsindəki şərt doğru olsun.
}
```

Yuxarıda göstərilən sintaksdakı {…} işarələr blok işarələridir. Bu o deməkdir ki, **if** mötərizəsindəki şərt doğru (**True**) olarsa, onda **if** bloku daxilindəki kodlar icra olunsun.
Qeyd: şərt hissəsində şərt operatorlarından (==, !=, >=, <=, >, <) istifadə olunur.
Gəlin **if** strukturunun blok-sxemlə təsvirinə baxaq.

Blok-sxemlə təsviri.



Blok-sxemlə təsvirdə **if** strukturunun işləmə prinsipi daha aydın nəzərə çarpır.

START – proqramın (və ya kodun) başlanması.
if (şərt) – şərtin **if** strukturu ilə yoxlanılması.
if bloku – əgər şərt **True** (doğru) olarsa, onda bu hissədəki kodlar icra olunacaq. Yox əgər şərt **False** (yanlış) olarsa, onda bu hissədəki kodlar icra olunmayacaq.
END – proqramın (və ya kodun) sonlanması.

Proqram (kod) nümunəsi 1.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 7;
    if(n == 7){
        cout<<"Yes";
    }
}
```

İzahı:

Bu proqramın çıxışında **Yes** sözü veriləcək. Çünki **n** dəyişəni 7-yə bərabər olduğu üçün **if** şərti **True** olacaq və **if** blokunun daxilindəki kod icra olunaraq çıxışa **Yes** sözü veriləcək.

Qeyd: Burada **n**-in başlanğıc qiyməti 7-dən fərqli ədəd olarsa, onda çıxışda **Yes** alınmayacaq.

Proqram (kod) nümunəsi 2.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    if(n == 4 || n == 5){
        cout<<"Yes";
    }
}
```

İzahı:

Bu proqramın çıxışında **Yes** sözü veriləcək. Bu koddakı **if** şərtində **or** (`||`) operatorundan istifadə etməklə iki şərtdən hər hansı birinin doğru olmasını yoxlayırıq, **n**-in qiyməti 5-ə bərabər olduğu üçün **if** şərti **True** olacaq və çıxışda **Yes** sözü veriləcək.

Qeyd: Burada **n**-in başlanğıc qiyməti 4 olsaydı yenə də çıxışda **Yes** olacaqdı. Lakin **n**-in dəyəri 4 və 5-dən fərqli ədəd olarsa, onda çıxışda **Yes** sözü alınmayacaq.

Proqram (kod) nümunəsi 3.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    if(n%2 == 1 && n == 5){
        cout<<"Yes";
    }
}
```

İzahı:

Bu proqramın çıxışında **Yes** sözü veriləcək. Bu koddakı **if** şərtində **and** (`&&`) operatorundan istifadə etməklə iki şərtdən hər ikisinin doğru olmasını yoxlayırıq, **n**-in qiyməti 5-ə bərabər və tək ədəd olduğu üçün **if** şərti **True** olacaq və çıxışda **Yes** sözü veriləcək.

Qeyd: Burada **n**-in başlanğıc qiyməti cüt ədəd və ya 5-dən fərqli ədəd olsaydı onda çıxışda **Yes** sözü alınmayacaqdı.

Proqram (kod) nümunəsi 4.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int x = 2, y = 3;
    if(x > 1){
        cout<<"A";
    }
    if(y < 4){
        cout<<"B";
    }
}
```

İzahı:

Bu proqramın çıxışında **AB** olacaq. Belə ki, **x>1** şərti **True** olduğu üçün ilk olaraq çıxışa **A**, sonra isə **y<4** şərti **True** olduğu üçün çıxışa **B** veriləcək.

Qeyd: Burada iki **if** şərti qeyd olunub və bu şərtlər bir-birindən asılı deyil, yəni ki, birinin **True** və ya **False** olması digərinin **True** və ya **False** olmasına təsir etmir.

Proqram (kod) nümunəsi 5.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int x = 2, y = 3;
    if(x > 1){
        cout<<"A";
        if(y < 4){
            cout<<"B";
        }
    }
}
```

İzahı:

Bu proqramın çıxışında **AB** olacaq. **x>1** şərti **True** olduğu üçün ilk olaraq çıxışa **A**, sonra isə **y<4** şərti **True** olduğu üçün çıxışa **B** veriləcək.

Qeyd: Burada iç-içə **if** (**nested if**) qeyd olunub. Diqqət etsək görərik ki, ikinci **if** birinci **if** bloku daxilində yazılıb. Əgər **x>1** şərti **False** olsaydı, onda **y<4** şərti yoxlanılmayacaqdı və nəticədə çıxışda heç bir dəyər alınmayacaqdı.

Proqram (kod) nümunəsi 6.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int x = 2, y = 3;
    if(x > y){
        cout<<"A";
        if(y < 4){
            cout<<"B";
        }
    }
    cout<<"C";
}
```

İzahı:

Bu proqramın çıxışında **C** olacaq. **x>y** şərti **False** olduğu üçün çıxışda **A** və **B** olmayacaq. Sonda çıxışda **C** dəyəri veriləcək, çünki bu dəyəri çıxışa verərkən heç bir şərtdən istifadə olunmayıb.

Qeyd: Burada iç-içə **if** (**nested if**) qeyd olunub. Diqqət etsək görərik ki, ikinci **if** birinci **if** bloku daxilində yazılıb və **x>y** şərti **False** olduğu üçün, **y<4** şərti yoxlanılmır və nəticədə çıxışda **A** və **B** dəyərlərinin heç biri olmur.

if-else strukturu

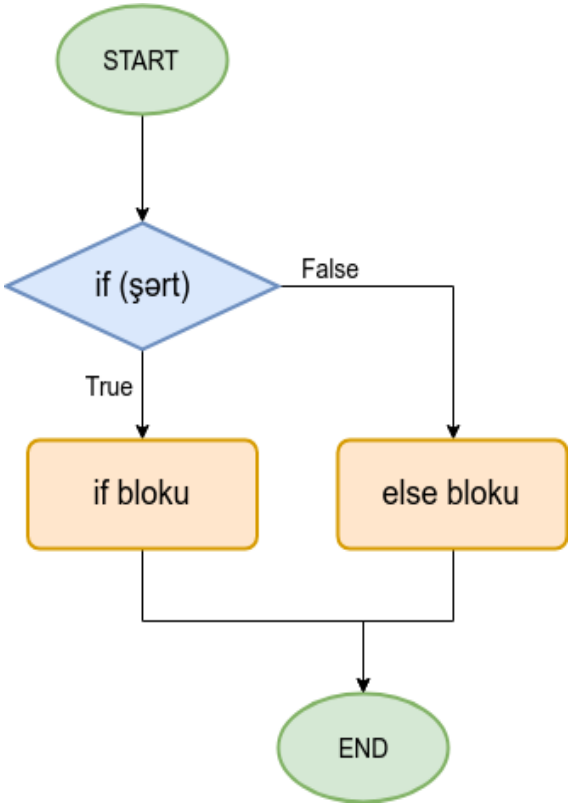
Bu struktur əsasında müəyyən şərtin doğru olması (**True**) ilə növbəti kod bloku icra olunur, əks halda (**False**) digər kod bloku icra olunur.

Sintaksı (quruluşu).

```
if(şərt)
{
      // if bloku
      // bu hissədəki kodlar o vaxt icra olunur ki,
      // if mötərizəsindəki şərt True olsun.
}
else{
      // else bloku
      // bu hissədəki kodlar o vaxt icra olunur ki,
      // if mötərizəsindəki şərt False olsun.
}
```

Yuxarıda göstərilən sintaksdakı {…} işarələr blok işarələridir. Bu o deməkdir ki, **if** mötərizəsindəki şərt doğru (**True**) olarsa, onda **if** bloku daxilindəki kodlar icra olunacaq. Əks halda, yəni ki şərt doğru olmazsa (**False**), onda **else** bloku daxilindəki kodlar icra olunacaq.
Gəlin **if-else** strukturunun blok-sxemlə təsvirinə baxaq.

Blok-sxemlə təsviri.



Blok-sxemlə təsvirdə **if-else** strukturunun işləmə prinsipi daha aydın nəzərə çarpır.

START – proqramın (və ya kodun) başlanması.
if (şərt) – şərtin **if** strukturu ilə yoxlanılması.
if bloku – əgər şərt **True** olarsa, onda bu hissədəki kodlar icra olunacaq.
else bloku – əgər şərt **False** olarsa, onda bu hissədəki kodlar icra olunacaq.
END – proqramın (və ya kodun) sonlanması.

Proqram (kod) nümunəsi 1.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    if(n > 0){
        cout<<"Yes";
    }
    else{
        cout<<"No";
    }
}
```

İzahı:

Bu proqramın çıxışında **Yes** sözü veriləcək. Çünki **n>0** şərti **True** olduğu üçün **if** blokunun daxilindəki kod icra olunaraq çıxışa **Yes** sözü veriləcək.

Qeyd: Burada **n** dəyişəni sıfırdan böyük olmasaydı onda **else** bloku daxilindəki kod icra olunaraq çıxışa **No** veriləcəkdi (Növbəti kod nümunəsinə baxın).

Proqram (kod) nümunəsi 2.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = -5;
    if(n > 0){
        cout<<"Yes";
    }
    else{
        cout<<"No";
    }
}
```

İzahı:

Bu proqramın çıxışında **No** sözü veriləcək. Çünki **n>0** şərti **False** olduğu üçün **else** blokunun daxilindəki kod icra olunaraq çıxışa **No** sözü veriləcək.

Proqram (kod) nümunəsi 3.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    if(n > 0 && n%2 == 0){
        cout<<"Yes";
    }
    else{
        cout<<"No";
    }
}
```

İzahı:

Bu proqramın çıxışında **No** sözü veriləcək. Bu koddakı **if** şərtində **and** (`&&`) operatorundan istifadə etməklə iki şərtdən hər ikisinin doğru olmasını yoxlayırıq, lakin **n**-in qiyməti cüt ədəd olmadığı üçün **if** şərti **False** olacaq və **else** bloku icra olunaraq çıxışda **No** sözü veriləcək.

Proqram (kod) nümunəsi 4.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    if(n > 0){
        cout<<"A";
    }
    if(n > 6){
        cout<<"B";
    }
    else{
        cout<<"C";
    }
}
```

İzahı:

Bu proqramın çıxışında **AC** veriləcək. Bu koddakı **else** bloku ona ən yaxın olan **if** blokuna aiddir. Yəni ki, **if(n>0)** blokunun **else** bloku ilə heç bir əlaqəsi yoxdur, lakin **if(n>6)** bloku isə **else** bloku ilə zəncirlənib. Beləliklə, **n>0** şərti **True** olduğu üçün çıxışda **A**, daha sonra isə **n>6** şərti **False** olduğu üçün **else** bloku icra olunaraq çıxışda **C** dəyəri veriləcək.

Proqram (kod) nümunəsi 5.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    if(n > 0){
        cout<<"A";
        if(n > 6){
            cout<<"B";
        }
    }
    else{
        cout<<"C";
    }
}
```

İzahı:

Bu proqramın çıxışında **A** veriləcək. Bu koddakı **else** bloku ona ən yaxın olan blokun **if**-nə aiddir. Diqqət etsək görərik ki, **else**-in üzərindəki qırmızı rəngli blok işarəsi **if(n>0)** blokunun bağlanışıdır və deməli **else** bloku ilə zəncirlənib, lakin **if(n>6)** blokunun isə **else** bloku ilə heç bir əlaqəsi yoxdur. Beləliklə, **n>0** şərti **True** olduğu üçün çıxışda **A** dəyəri veriləcək, yəni ki **else** bloku icra olunmayacaq. **if(n>6)** şərti isə **False** olduğu üçün çıxışa **B** dəyəri verilməyəcək.

Proqram (kod) nümunəsi 6.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 5;
    else{
        cout<<"Yes";
    }
    if(n > 0){
        cout<<"No";
    }
}
```

İzahı:

Bu kod kompilyasiya xətası verəcək (**Error**). Bu cür sintaks yazılışı tamamilə səhvdir. **else** bloku hər zaman **if** şərtindən sonra gəlməlidir. Yəni ki, **else** bloku özündən əvvəlki **if** bloku ilə zəncirlənməlidir.

if-else-if strukturu

Bu struktur əsasında müəyyən şərtin doğru (**True**) olması ilə növbəti kod bloku icra olunur, əks halda (**False**) digər kod bloku yoxlanılır və beləcə **True** şərti olana qədər davam edir. İlk tapılan hər hansı **True** şərti olarsa, həmin kod bloku icra olunur və digər şərtlərə baxılmır.

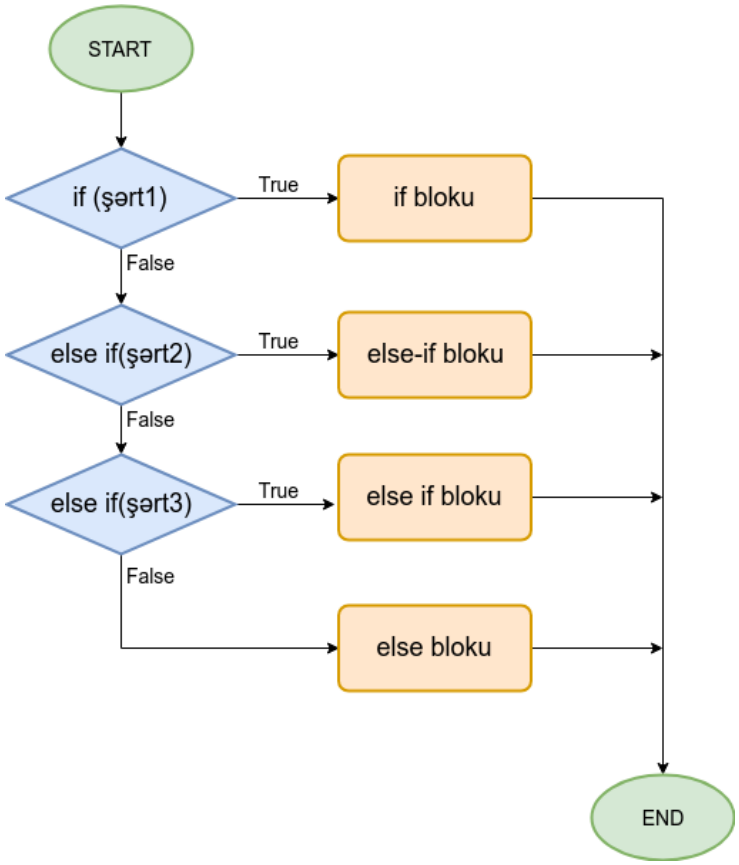
Sintaksı (quruluşu).

```
if(şərt1)
{
     // if bloku
     // bu hissədəki kodlar o vaxt icra olunur ki,
     // if mötərizəsindəki şərt True olsun.
}
else if(şərt2){
     // else-if bloku
     // bu hissədəki kodlar o vaxt icra olunur ki,
     // şərt1 False olsun və mötərizəsindəki şərt2
     // True olsun.
}
else if(şərt3){
     // else-if bloku
     // bu hissədəki kodlar o vaxt icra olunur ki,
     // şərt1 və şərt2 False olsun,
     // lakin şərt3 True olsun
}
/*
  else-if bloklarının sayında məhdudiyyət yoxdur.
  Verilən məsələdən aslı olaraq istənilən sayda
  else-if blokları yaza bilərik.
*/
else{
     // else bloku
     // bu hissədəki kodlar o vaxt icra olunur ki,
     // əvvəlki şərtlərin hamısı False olsun.
}
```

Yuxarıdakı sintaksdakı bloklardan yalnız biri icra oluna bilər. **şərt1 True** olarsa, növbəti şərtlərə baxılmır. **şərt1 False**, lakin **şərt2 True** olarsa, onda **şərt2**-dən sonra gələn şərtlərə baxılmayacaq. Bu struktur bu şəkildə davam edəcək və ən sonda heç bir şərt **True** olmazsa, onda **else** bloku icra olunacaq.

Gəlin **if-else-if** strukturunun blok-sxemlə təsvirinə baxaq.

Blok-sxemlə təsviri.



Blok-sxemlə təsvirdə **if-else-if** strukturunun işləmə prinsipi daha aydın nəzərə çarpır.

START – proqramın (və ya kodun) başlanması.
if (şərt1) – şərt1-in **if** strukturu ilə yoxlanılması.
if bloku – əgər şərt1 **True** (doğru) olarsa, onda bu hissədəki kodlar icra olunacaq və sonrakı şərtlərə baxılmayacaq.
else if (şərt2) – şərt2-in **else-if** strukturu ilə yoxlanılması.
else-if bloku – əgər şərt1 **False** və **şərt2 True** olarsa, onda bu hissədəki kodlar icra olunacaq və sonrakı şərtlərə baxılmayacaq.
else if (şərt3) – şərt3-ün **else-if** strukturu ilə yoxlanılması.
else-if bloku – əgər **şərt1** və **şərt2 False** və **şərt3 True** olarsa, onda bu hissədəki kodlar icra olunacaq və sonrakı şərtlərə baxılmayacaq.
else bloku – **şərt1**, **şərt2** və **şərt3 False** olarsa, onda **else** blokundakı kodlar icra olunacaq.
END – proqramın (və ya kodun) sonlanması.

Proqram (kod) nümunəsi 1.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 0;
    if(n > 0){
        cout<<"Positive";
    }
    else if(n == 0){
        cout<<"Zero";
    }
    else{
        cout<<"Negative";
    }
}
```

İzahı:

Bu proqramın çıxışında **Zero** sözü veriləcək. **if(n>0)** şərti **False** olduğu üçün növbəti **else if(n==0)** şərti yoxlanılaraq **True** olacaq və çıxışda **Zero** sözü veriləcək.

Proqram (kod) nümunəsi 2.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int n = 12;
    if(n%2 == 0){
        cout<<"A";
    }
    else if(n%3 == 0){
        cout<<"B";
    }
    else if(n%4 == 0){
        cout<<"C";
    }
}
```

İzahı:

Bu proqramın çıxışında yalnız **A** veriləcək. **if(n%2==0)** şərti **True** olduğu üçün növbəti şərtlərin **True** və ya **False** olmağından aslı olmayaraq heç biri yoxlanılmayacaq.

Qeyd: Burada digər şərtlərin hamısı **True** nəticəsini verir, lakin heç biri icra olunmur. Həmçinin ən sonda **else** bloku yoxdur, bu o deməkdir ki, **if-else-if** zəncirində ən sonda **else** bloku yazılmaya da bilər (bu verilən məsələdən aslıdır).

Proqram (kod) nümunəsi 3.

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    int day = 3;
    if(day == 1){
        cout<<"Monday";
    }
    else if(day == 2){
        cout<<"Tuesday";
    }
    else if(day == 3){
        cout<<"Wednesday";
    }
    else if(day == 4){
        cout<<"Thursday";
    }
    else if(day == 5){
        cout<<"Friday";
    }
    else if(day == 6){
        cout<<"Saturday";
    }
    else if(day == 7){
        cout<<"Sunday";
    }
    else{
        cout<<"Not a day";
    }
}
```

İzahı:

Bu kod həftənin hansı gün olduğunu sözlə çıxışa verir. **day** dəyişənində həftənin rəqəmlə gününü yazırıq və çıxışda həmin rəqəmin həftənin uyğun gününə bərabərliyini yoxlayaraq çıxışa həftənin gününü sözlə veririk.

Bu proqramın çıxışında **Wednesday** sözü veriləcək. **if(day==3)** şərti **True** olduğu üçün, və ondan öncəki şərtlərin hamısının **False** olduğu üçün çıxışda yalnız **Wednesday** sözü veriləcək, və bundan sonra gələn **else-if** bloklarının heç bir şərtləri yoxlanılmayacaq.

Qeyd: Bu proqramda ən sonda **else** bloku istifadə etmişik, bunun məqsədi **day** dəyişəninin **[1;7]** intervalından kənarda olduğunu yoxlamaqdır. Əgər **day** dəyişəni intervaldan kənara çıxarsa onda həmin ədədin heç bir günə uyğun gəlmədiyini **Not a day** ifadəsi ilə çıxışda bildiririk.