

Introduction to Java

حلقه ها

Controlling Program Flow

The objectives of this chapter are:

- To explain the three Java looping mechanisms
 - for
 - while
 - do while

Loops

- Loops are used to execute statements or blocks multiple times based on a looping condition.
- Java has three types of loops:
 - while loops
 - do while loops
 - for loops
- Care should be taken whenever a loop is used to avoid an endless loop.

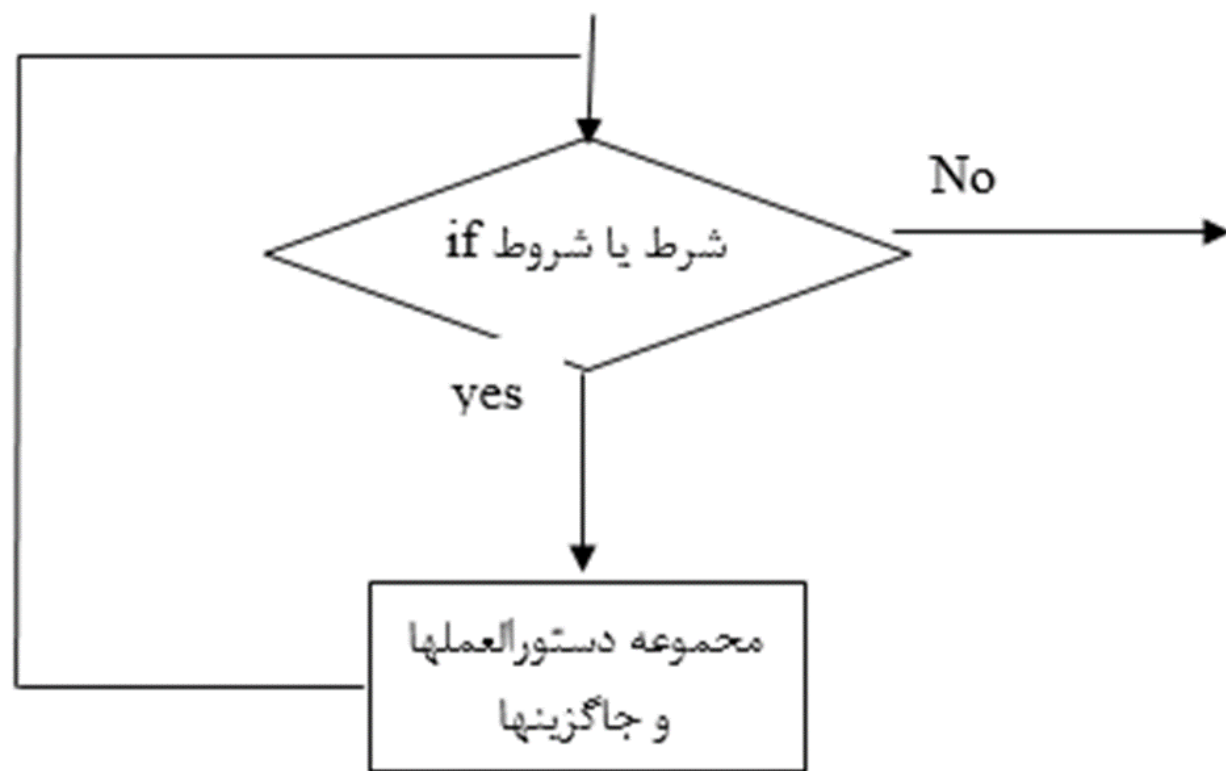
while loops

- The while loop is the most basic loop in Java.

```
while (boolean-expression)
{
    statement1;
    [...]
}
```

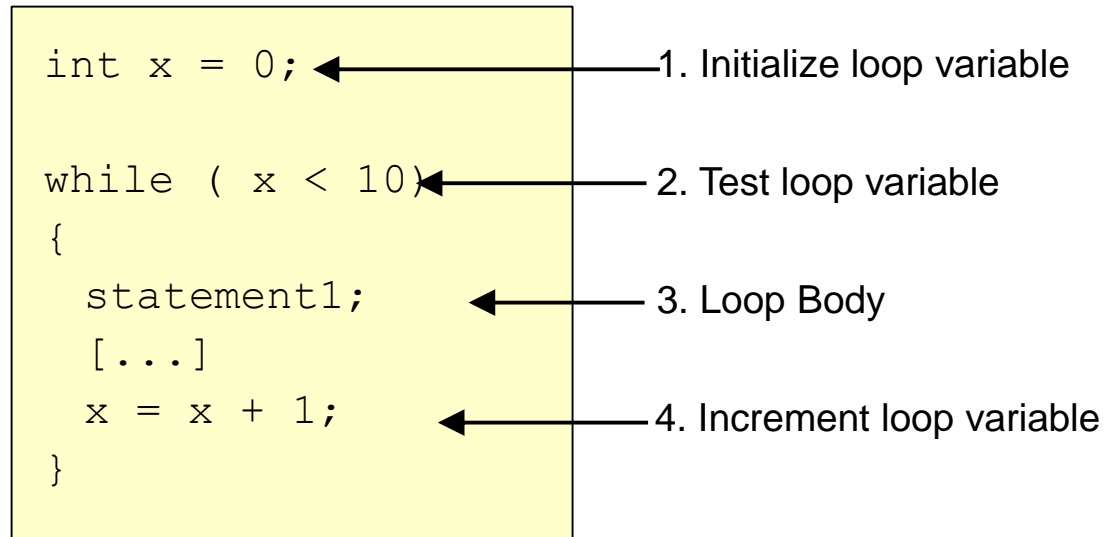
- The loop body will continue to execute as long as the looping condition is true. The looping condition is tested upon entry and when the loop body is completed.
- If the loop body consists of a single statement, the curly braces are not necessary.
- If the looping condition is false upon entry, the loop body will not be executed.

در این حلقه‌ها با توجه به ورودی، تعداد تکرار مشخص می‌شود. و دقیقاً نمی‌توان تعداد تکرار حلقه را بدون ورودی معین کرد. این حلقه‌ها فقط شامل شرطی یا شروطی هستند که تا زمانیکه آنها برقرار باشند حلقه اجرا می‌شود. در حالت کلی این نوع حلقه‌ها بصورت زیر نمایش داده می‌شوند:



Loop Components

- Each loop has 4 main components



Loops - Examples

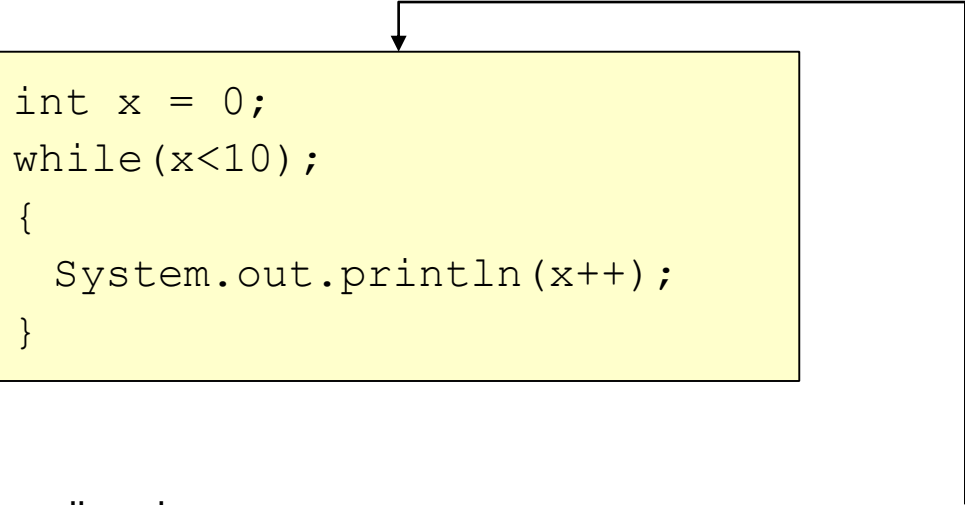
- What will these loops output?

```
int x = 0;
while(x<10)
{
    System.out.println(x++);
}
```

```
int x = 0;
while(x<10)
{
    System.out.println(++x);
}
```


Loops - Common Errors

- What is the error in this loop?



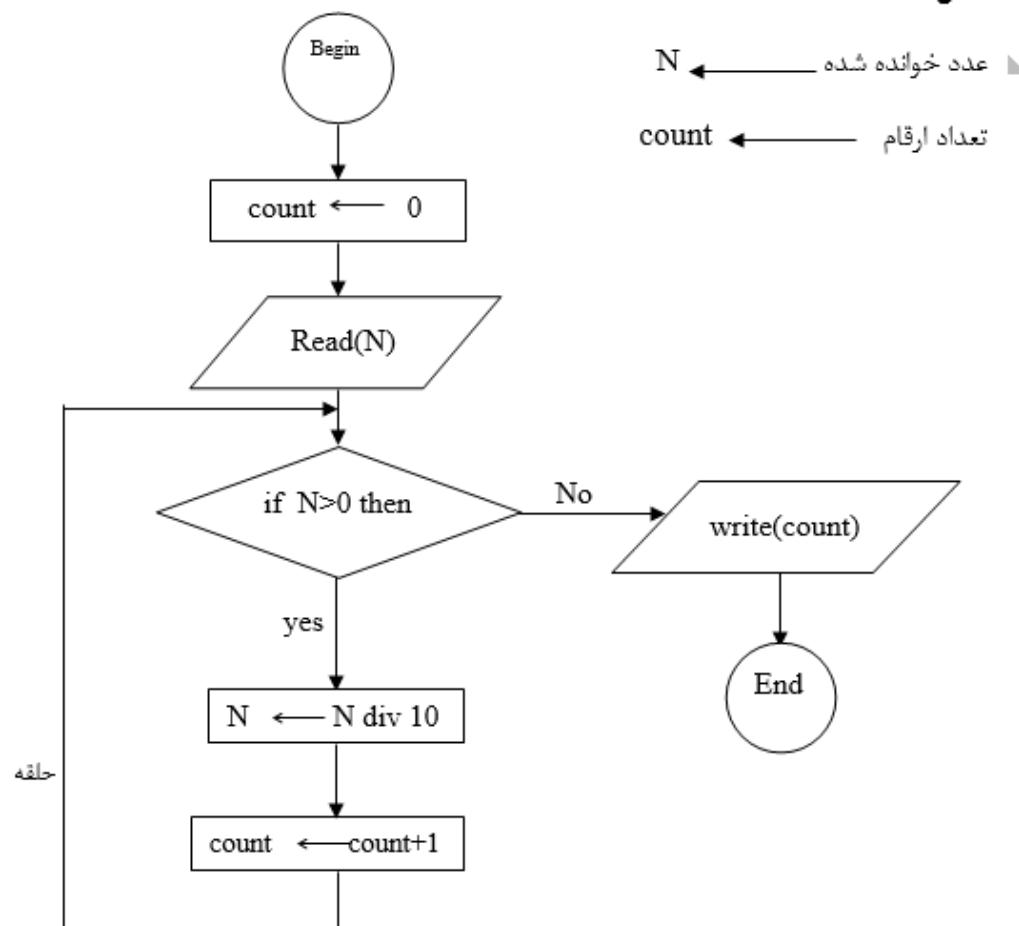
```
int x = 0;
while(x<10);
{
    System.out.println(x++);
}
```

The diagram shows a yellow box containing the code. A black arrow originates from the bottom of the box, goes down, then right, then up, and finally left, pointing back to the top of the box. This visualizes the loop's execution path, which never reaches a termination point.

endless loop

حلقه بینهایت

مثال: فلوچارتی رسم کنید که عددی را از ورودی دریافت کرده سپس تعداد ارقام آن را شمرده در خروجی چاپ نماید.



مثال برنامه ای بنویسید که عددی از ورودی دریافت کرده، سری فیبوناچی قبل از آنرا تولید نماید.

سری بصورت زیر می باشد :

0 1 1 2 3 5 8 13 ...

و در حالت کلی جملات سری بصورت:

$$f_k = f_{k-1} + f_{k-2}$$

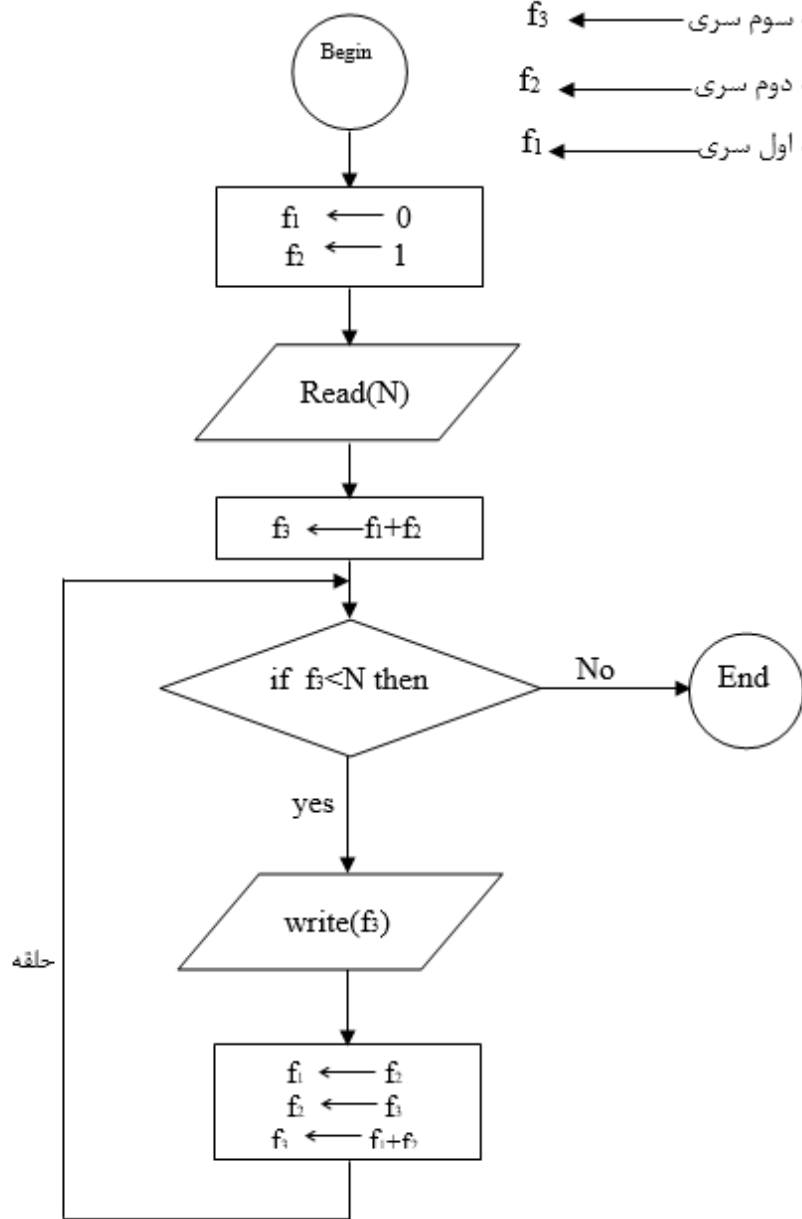
مثال

عدد خوانده شده $\leftarrow N$

جمله سوم سری $\leftarrow f_3$

جمله دوم سری $\leftarrow f_2$

جمله اول سری $\leftarrow f_1$



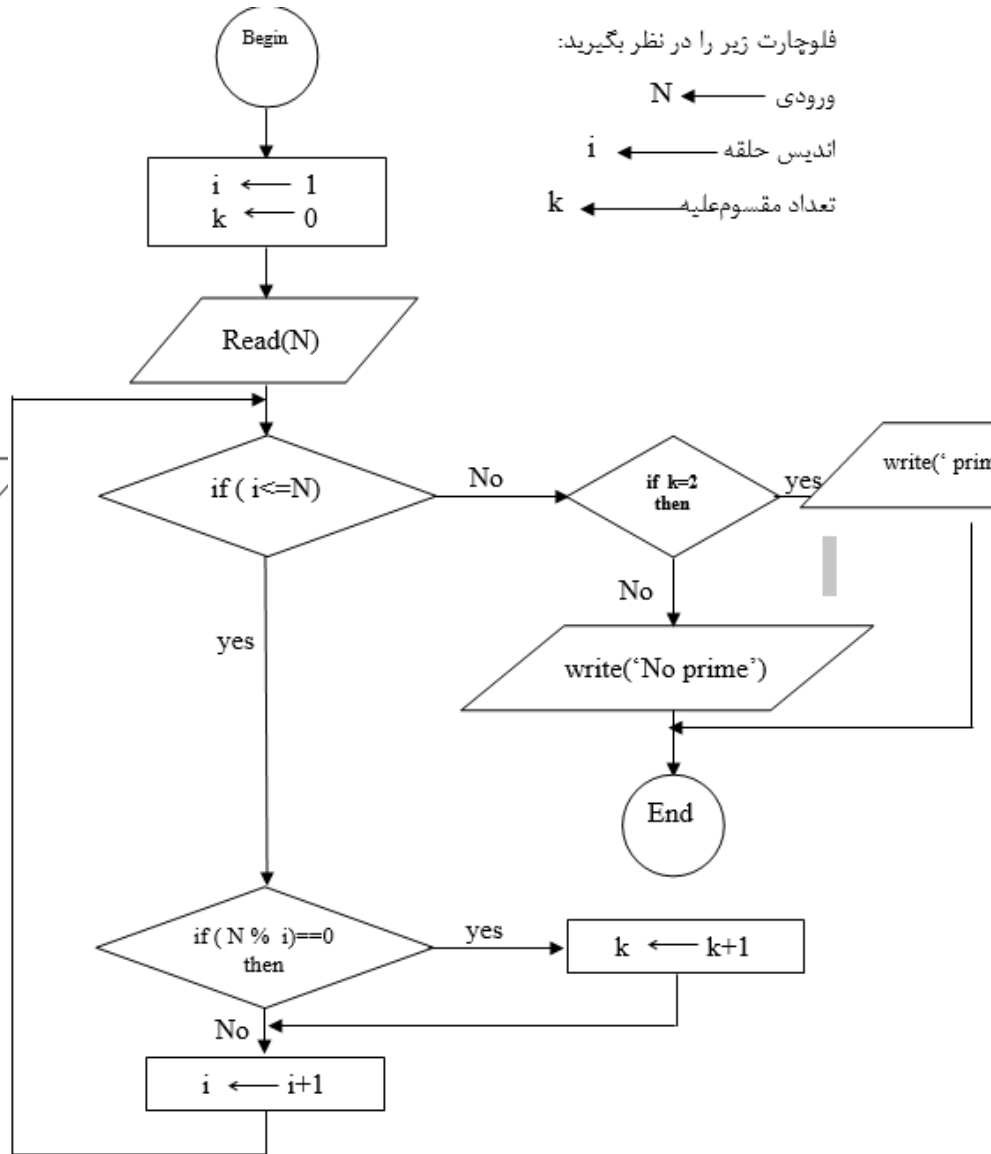
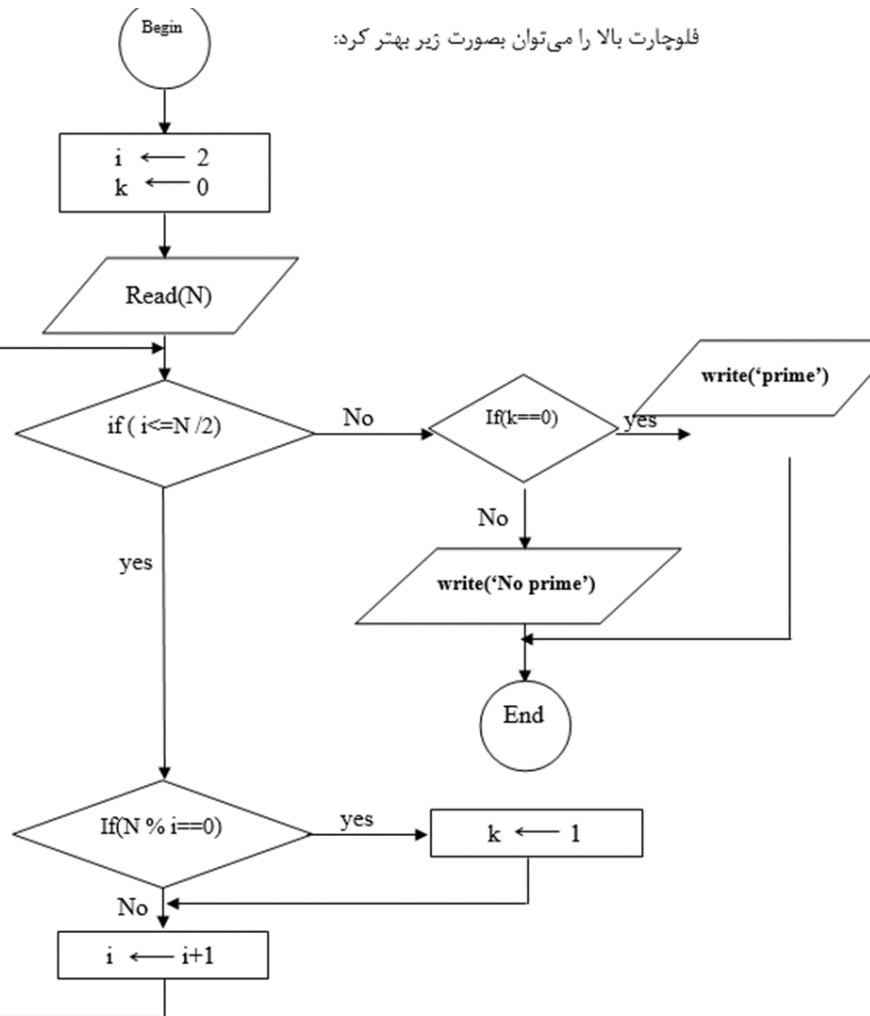
برنامه ای بنویسید که عددی از ورودی دریافت کرده، اول بودن عدد را بررسی نماید.

فلوچارت زیر را در نظر بگیرید:

ورودی N

اندیس حلقه i

تعداد مقسوم علیه k



- برنامه ای بنویسید که مقلوب عدد را محاسبه و چاپ نماید.

- فلوچارتی رسم نمائید که دو عدد M , N را از ورودی خوانده، بزرگترین مقسوم علیه مشترک دو عدد را محاسبه و چاپ کند.

مثال

- برنامه ای بنویسید که N عدد از ورودی دریافت کرده، بزرگترین مقدار از بین N عدد و تعداد تکرار آن را محاسبه و چاپ نماید.
- برنامه ای بنویسید که عددی از ورودی خوانده، آن را به مبنای ۲ ببرد.
- برنامه ای بنویسید که، عدد N را از ورودی خوانده، تشخیص دهد عدد خوانده شده فاکتوریل چه عددی است.

حلقه for

در این نوع حلقه‌ها تعداد تکرار مشخص می‌باشد و هدف حلقه تکرار تعدادی از دستورالعمل‌ها به تعداد معین می‌باشد. این حلقه از اجزاء زیر تشکیل می‌شود:

- اندیس حلقه
- مقدار اولیه برای اندیس حلقه
- مقدار افزاینده برای اندیس حلقه (معمولا یک واحد در هر مرحله)
- مقدار نهایی (تعداد تکرار حلقه)
- شرطی برای کنترل تعداد تکرار حلقه

for loop

- The syntax of the for loop makes the 4 parts of the loop explicit.

```
int x = 0; ← 1. Initialize loop variable  
  
while ( x < 10) ← 2. Test loop variable  
{  
    statement1; ← 3. Loop Body  
    [...]   
    x = x + 1; ← 4. Increment loop variable  
}
```

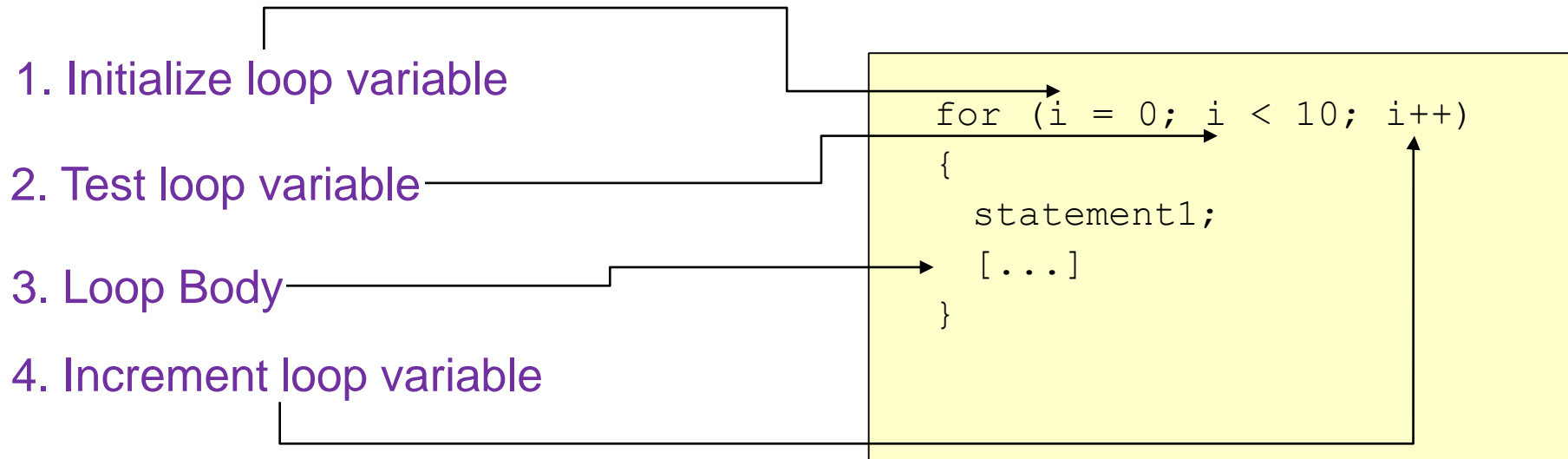
```
for (i = 0; i < 10; i++)  
{  
    statement1;  
    [...]   
}
```

syntax:

for loop

- The syntax of the for loop makes the 4 parts of the loop explicit.


syntax:



for loop

■ Examples

```
int x=0;
for (int i = 0; i < 10; i++)
{
    System.out.println(x++);
    [...]
}
```



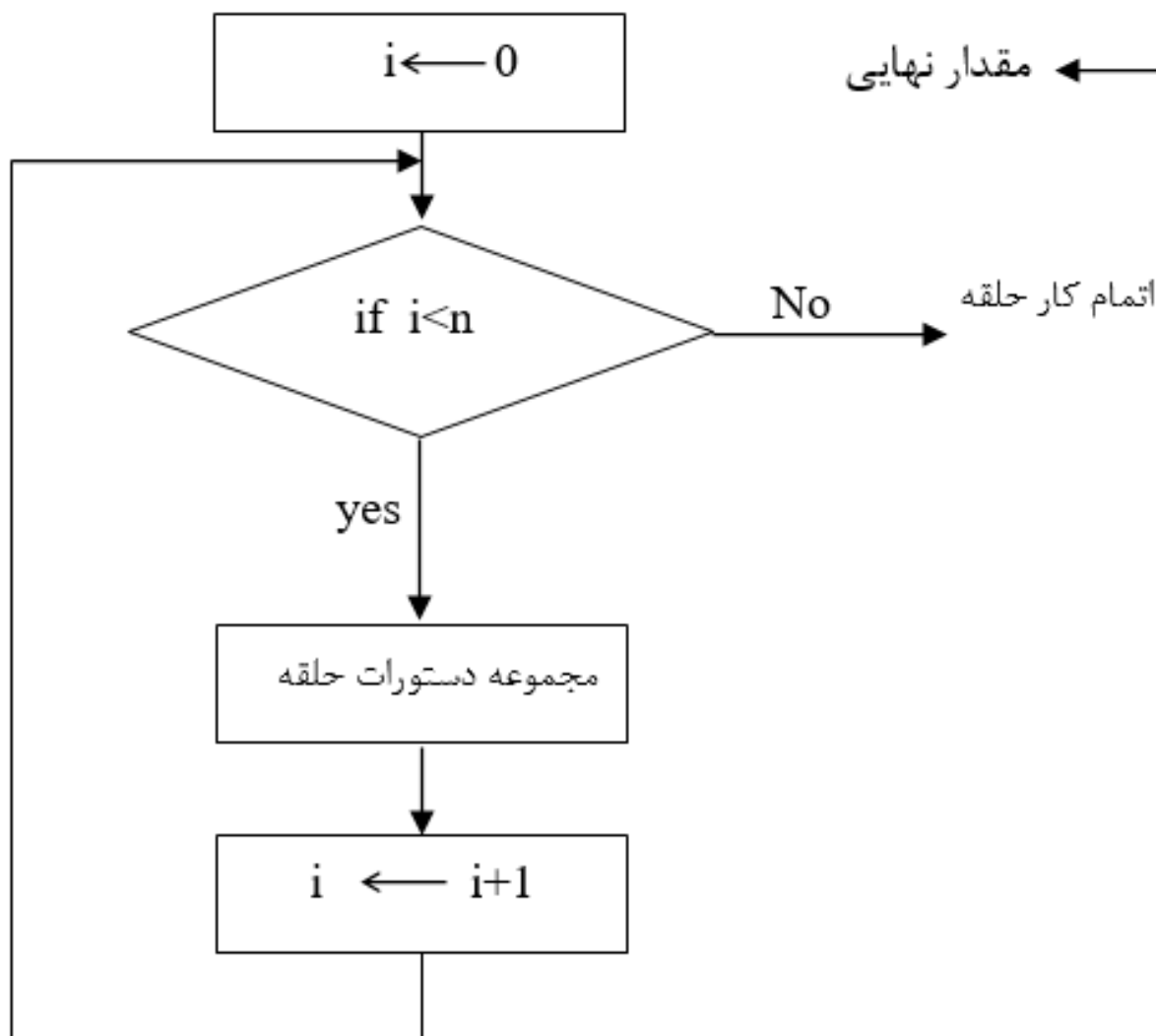
```
int x=0;
for (int i = 0; i < 10; i++);
{
    System.out.println(x++);
    [...]
}
```

مثال

این حلقه‌ها را غالباً با فلوجارت بصورت زیر نمایش می‌دهند:

$i \leftarrow$ اندیس حلقه

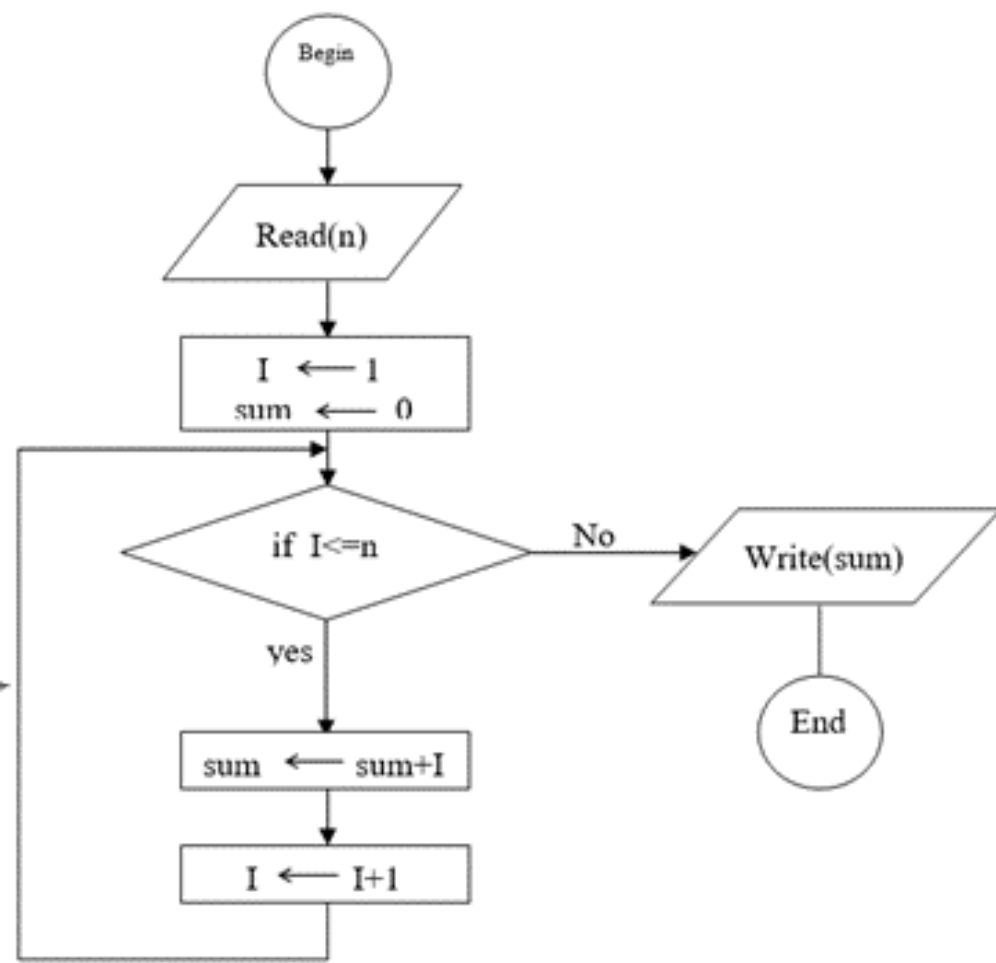
$n \leftarrow$ مقدار نهایی



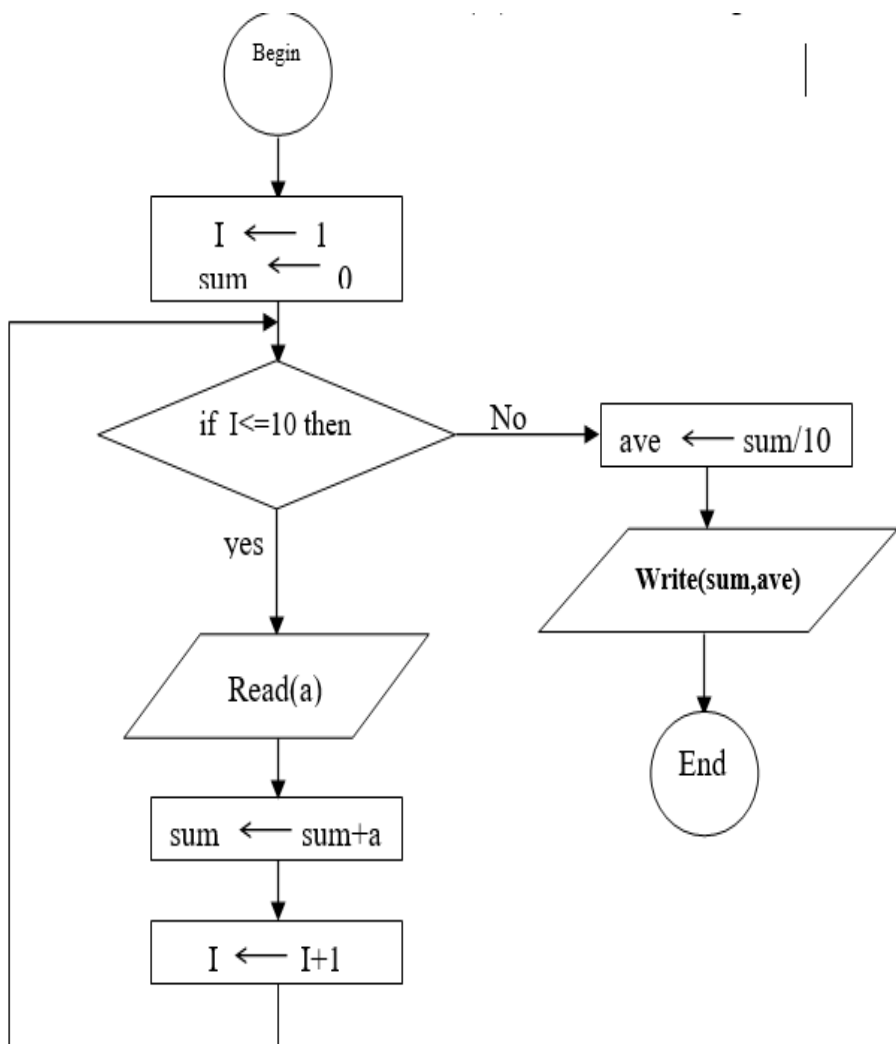
فلوچارتی رسم نمائید که عدد n را از ورودی دریافت کرده، مجموع اعداد از یک تا n را محاسبه کند.

اندیس حلقه \longrightarrow i

مقدار نهایی \longrightarrow n

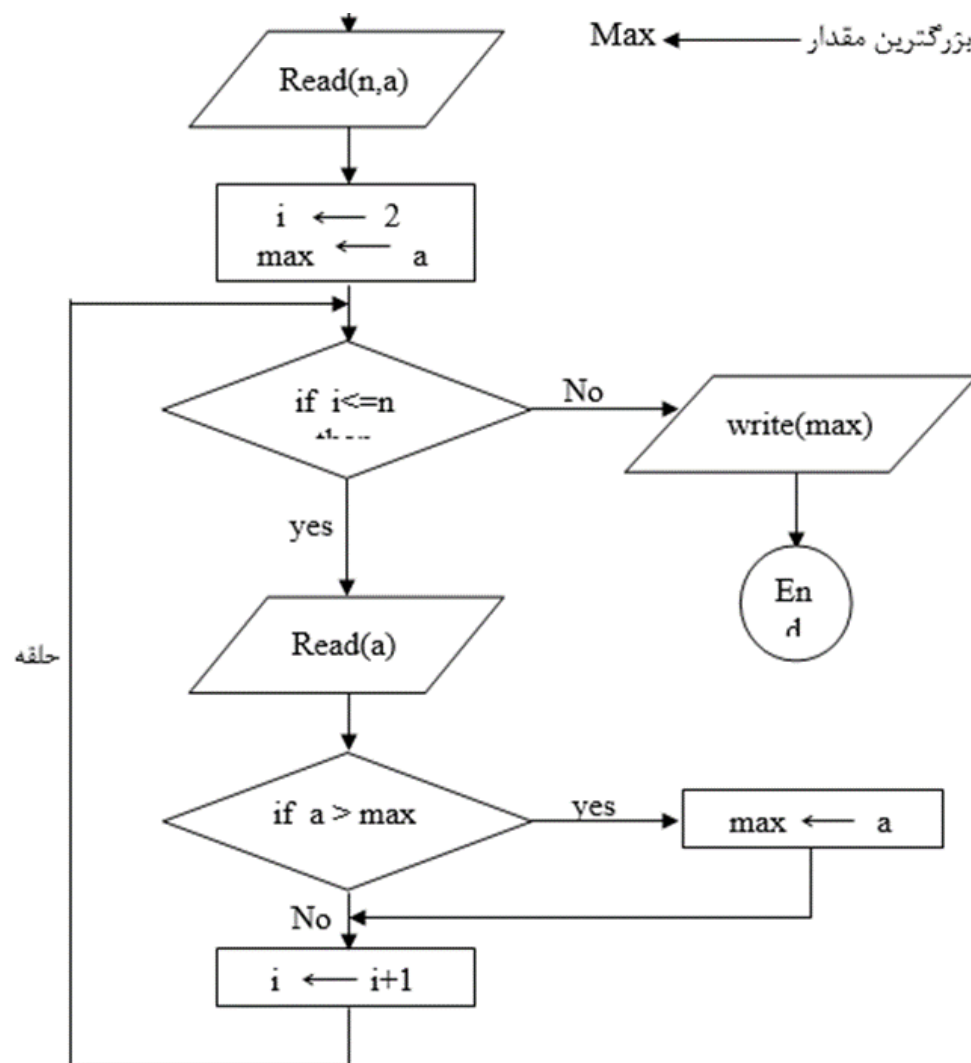


برنامه ای بنویسید که که ۱۰ عدد از ورودی دریافت کرده، مجموع و میانگین ۱۰ عدد را محاسبه و چاپ کند.



مثال

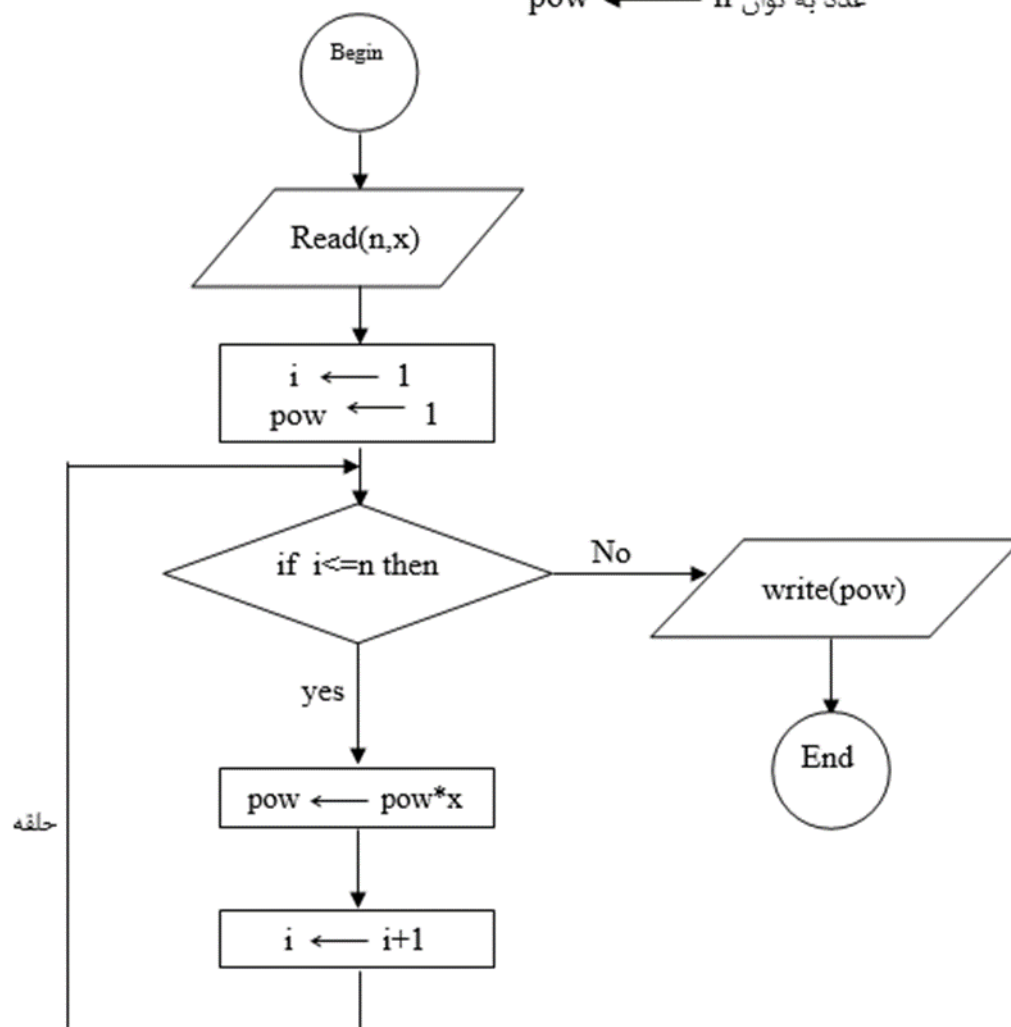
فلوچارتی رسم کنید که n عدد از ورودی دریافت کرده، بزرگترین مقدار از بین n عدد را پیدا کرده در خروجی چاپ نماید.



مثال

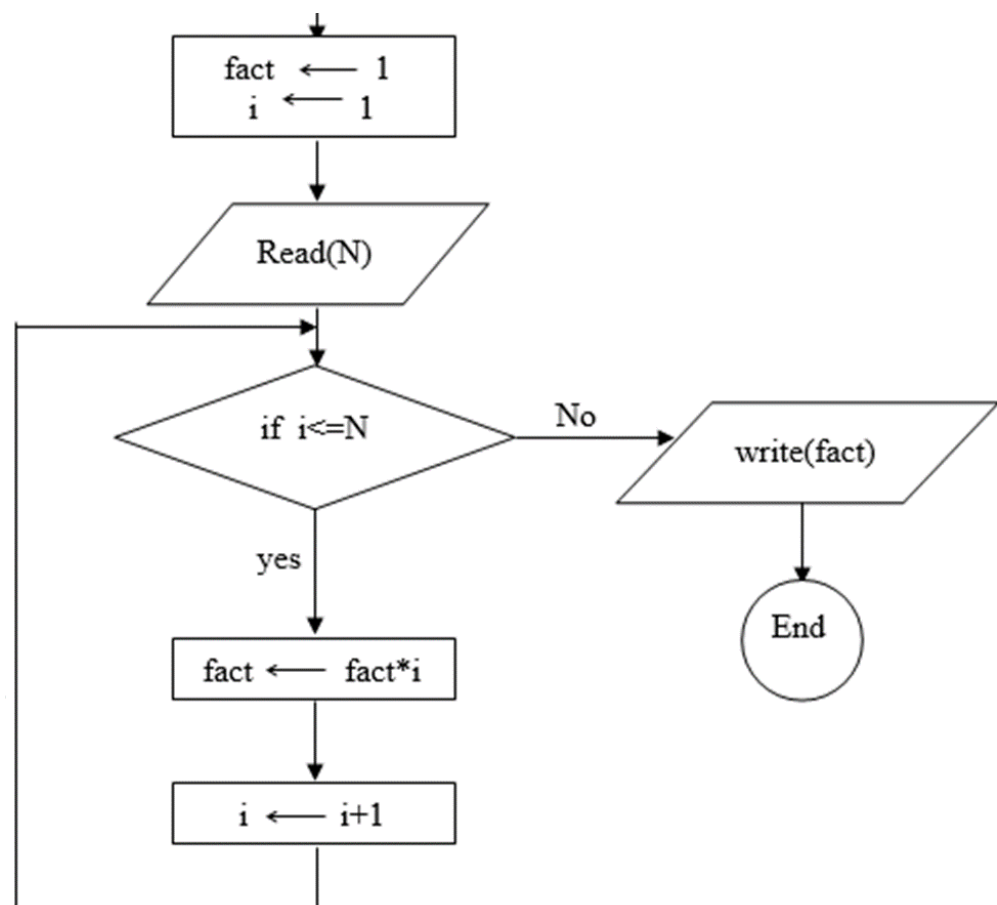
:فلوچارتی رسم نمائید که n و x ، دو عدد صحیح مثبت را از ورودی دریافت کرده سپس x به توان n را محاسبه کند.

عدد به توان $n \leftarrow \text{pow}$



مثال

فلوچارتی رسم کنید که عدد N را از ورودی دریافت کرده، فاکتوریل آنرا محاسبه نماید.



do-while

```
do
{
    state1..n;
} while (conditions);
```

```
int x = 0;
do
{
    System.out.println(x);
    x = x+1;
} while (x<10);
```

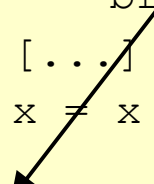
```
int x = 0;
do
{
    System.out.println(x);
} while (x<10);
```

break revisited

- We previously saw the break statement used in switch statements.
 - break can also be used with loops.
- The break statement will cause the flow of execution to break out of the current loop.
- If loops are nested, break will cause control to leave the inner-most loop.

```
int x = 0;

while ( x < 10)
{
    if (y > 100)
        break;
    [...]
    x = x + 1;
}
```



Example

```
int x=0;
for(int i=0;i<10;i++)
    for(int j=0;j<5; j++)
    {
        System.out.println(x++);
        if(i%2==0)
            break;
    }
```

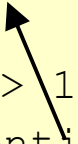
continue

- continue is similar to break.
- continue causes execution to go back to the loop test condition. If the test condition is true, the loop will be executed again. If not, the loop body is exited.

```
int x = 0;

while ( x < 10)
{
    if (y > 100)
        continue;

    [...]
    x = x + 1;
}
```



حلقه های تودرتو

الگوریتم‌هایی که تا حال بکار بردیم، فقط شامل یک حلقه بودند.

در صورتی که در بسیاری از مسائل ممکن است نیاز به استفاده از چند حلقه در داخل هم باشیم. در این نوع حلقه‌ها باید دقت بیشتری به خرج دهیم، تا مشکلی پیش نیاید. اگر از حلقه‌های نوع اول بصورت تودرتو استفاده کنیم در اینصورت برای هر حلقه شرط نهایی و اندیس اولیه جداگانه باید تعریف کنیم. در حلقه‌های تودرتو به ازای یکبار تکرار حلقه اولیه، حلقه داخلی به اندازه مقدار نهایی خود تکرار می‌شود. در کل اگر حلقه اولیه n بار تکرار شود و حلقه داخلی m بار، در اینصورت کل حلقه $n \times m$ بار تکرار خواهد شد.

فلوچارت حلقه‌های تودرتو را می‌توان بصورت زیر نشان داد:

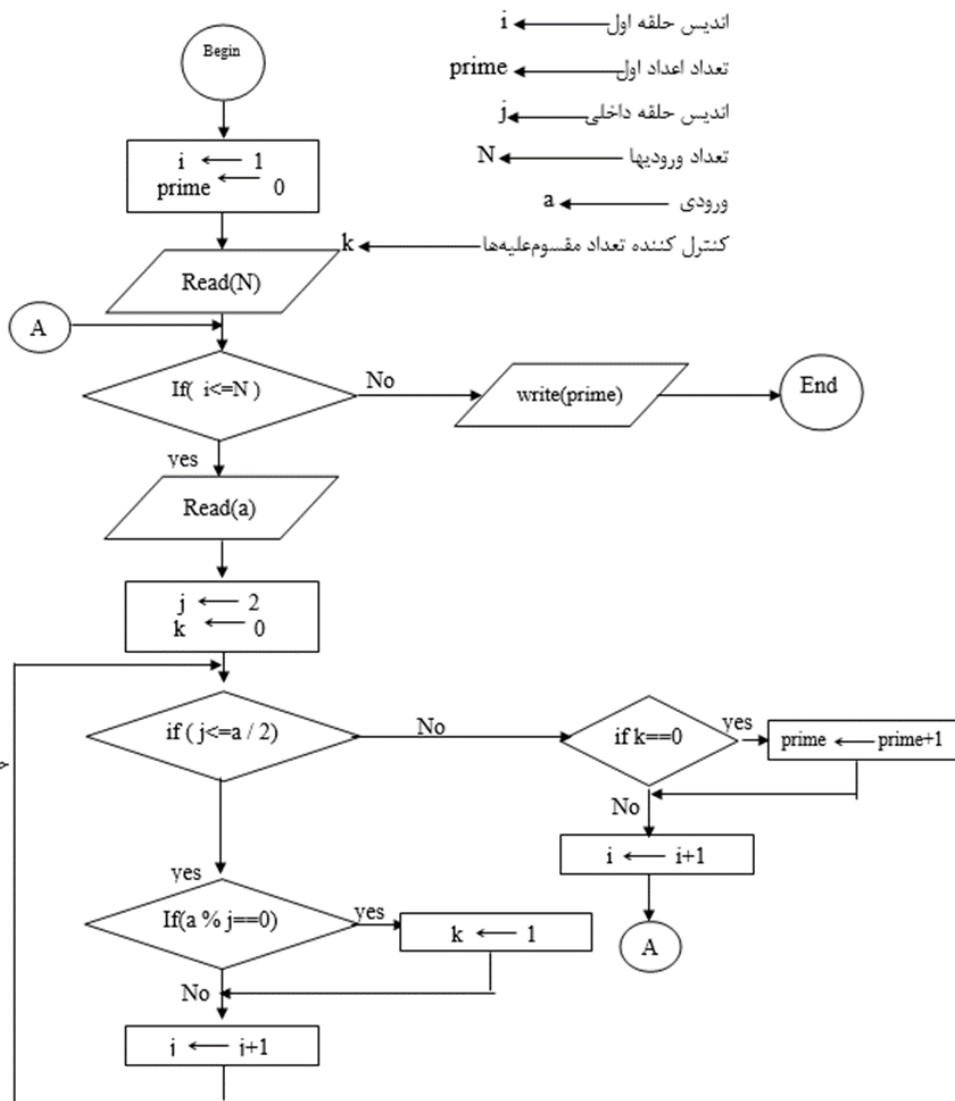
اندیس حلقه اول $\leftarrow i$

اندیس حلقه داخلی $\leftarrow j$

مقدار نهایی حلقه اول $\leftarrow n$

مقدار نهایی حلقه داخلی $\leftarrow m$

مثال فلوچارتی رسم نمائید که N عدد از ورودی دریافت کرده تعداد اعداد اول را شمرده در خروجی چاپ نماید.



مثال

برنامه ای بنویسید که N را از ورودی دریافت کرده، مجموع سری زیر را محاسبه نماید:

$$S=1+\frac{2}{2!}+\frac{3}{3!}+\dots+\frac{N}{N!}$$

