Fundamentals of Loop and Iteration in Algorithms

Sometimes, we need to execute a set of commands multiple times to solve a problem. Instead of typing the same commands multiple times, we can use the recursive command, which allows us to “write a set of commands once” and “execute multiple times.” Repetition is often referred to as "loops". Redundancy provides the ability to reuse instructions and simplifies problem-solving steps.

في بعض الأحيان، نحتاج لتنفيذ مجموعة من الأوامر عدة مرات لحل مشكلة ما. فعوض أن نكتب نفس الأوامر عدة مرات يمكن أن نستعين بالأمر التكراري، والذي يسمح لنا "بكتابة مجموعة من الأوامر مرة واحدة" و"التنفيذ عدة مرات". و يُشار إلى التكرار غالبًا باسم **"الحلقات"** . توفر الحلقات إمكانية إعادة استخدام التعليمات عدة مرات ويبسط خطوات حل المشكلات.

Types of loops in Algorithmic

Iteration is implemented using the loop in programming, where we primarily use two types of loops: "**for**" loop and "**while**" loop.

for loop in Algorithmic

We use a **for** loop when we know the number of times the loop will be executed. In other words, the for loop helps us execute a sequence of instructions a predetermined .number of times

To write a **for** loop, we use the loop variable to control the number of times the loop is executed. We assign the initial value of the variable to the start point of the loop, followed by the final value to be the end point of the loop. The code inside the loop will be executed, and after each execution, the value of the loop variable is automatically incremented and we return Repeat this step until the value of the loop variable reaches the final value.

نستخدم حلقة **من أجل (for)**عندما نعرف عدد مرات تنفيذ الحلقة. بمعنى آخر، تساعدنا حلقة **for** على تنفيذ سلسلة تعليمات بعدد مرات محدد مسبقا .

لكتابة الحلقة **for** ، نستخدم متغير الحلقة للتحكم في عدد مرات تنفيذ الحلقة، حيث نسند القيمة الأولية للمتغير نقطة بداية الحلقة، تليها القيمة النهائية نقطة نهاية الحلقة، فسيتم تنفيذ التعليمات البرمجية الموجودة داخل الحلقة و بعد كل تنفيذ، يتم زيادة قيمة متغير الحلقة آليا و نعيد تكرار هذه الخطوة حتى تصل قيمة متغير الحلقة الى القيمة النهائية .

for (I ← init-value , final value, step value) **do**
  Loop **body**
end **for**

While loop in programming

The while loop is used to execute the loop body until a specific condition is **false**. We mainly apply this idea when we don't know how many times the loop will execute.

The while loop consists of a loop condition, a block of code as a loop body, and a loop update expression if required. First, the loop condition is evaluated, and if it is true, code within the loop body will be executed. This process repeats until the loop condition becomes false. For better intuition, while loop can be thought of as a repeating if statement.

يتم استخدام حلقة **while** لتنفيذ أوامر داخل الحلقة حتى يصبح الشرط المنطقي لـ **while** خاطئًا. نستخدم الحلقة **while** بشكل أساسي عندما يكون عدد التكرارات اللازم تنفيذها في الحلقة مجهول (متعلق بشرط) .

تتكون الحلقة **while** من شرط حلقة، وسلسلة التعليمات البرمجية داخل الحلقة تحتوي على تعليمات لتحديث شرط الحلقة. أولاً يتم معاينة شرط الحلقة، وإذا كان صحيحا، فسيتم تنفيذ التعليمات البرمجية الموجودة داخل الحلقة. تتكرر هذه العملية حتى يصبح شرط الحلقة خاطئ.

initialisation **cond**
while (loop condition) do
    loop body
    cond update
end **while**

Special Notes

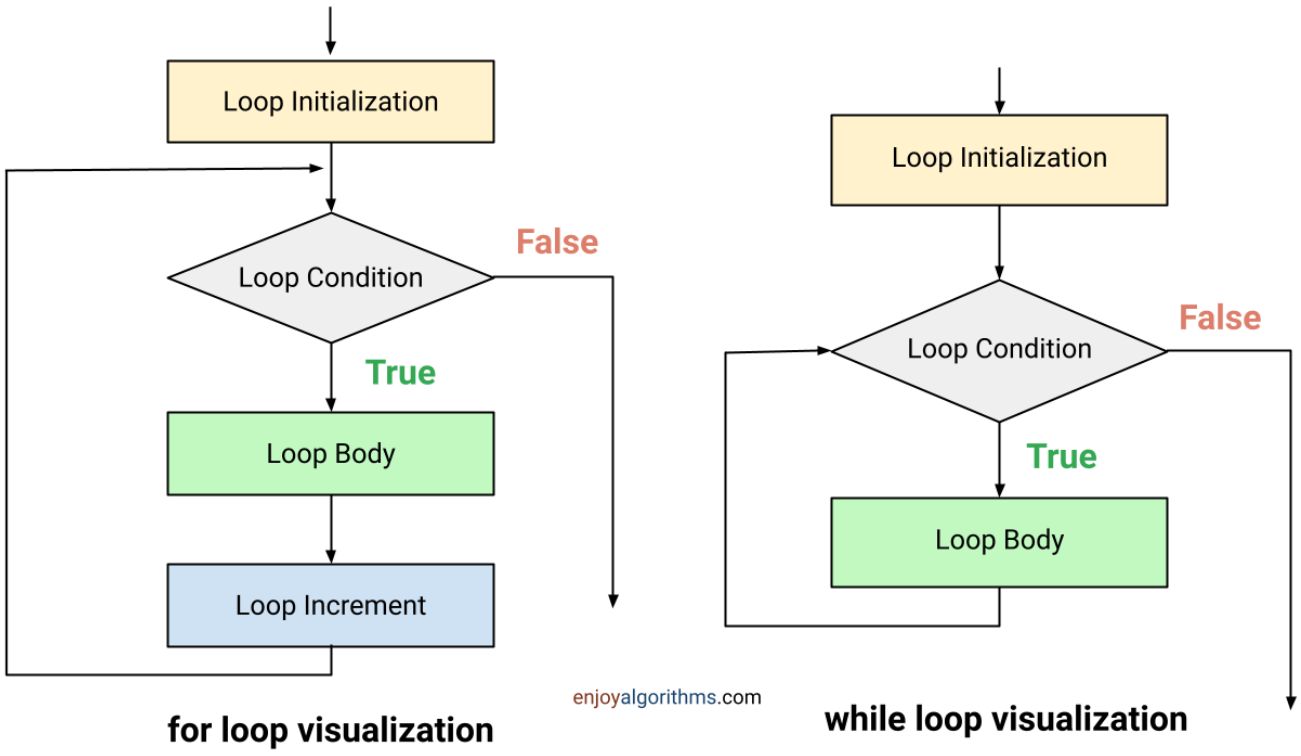
- There is also a third type of loop in programming: do-while loop. It is similar to while loop, where loop execution is terminated based on some condition. But the do-while loop condition is tested at the end of loop body.
- The do-while loop is exit controlled, whereas the for and while loops are entry-controlled loops.
- In the do-while loop, loop body will execute at least once irrespective of the test condition.

ملاحظات

- هناك أيضًا نوع ثالث من الحلقات في البرمجة: حلقة **do-while**. وهي تشبه حلقة while، حيث يتم إنهاء تنفيذ الحلقة بناءً على بعض الشروط. ولكن يتم اختبار شرط الحلقة do-while في نهاية جسم الحلقة.

• حلقة do-while يتم التحكم فيها عند الخروج، في حين أن حلقات for و while عبارة عن حلقات يتم التحكم فيها عند الدخول.

• في حلقة do-while، سيتم تنفيذ جسم الحلقة مرة واحدة على الأقل بغض النظر عن شرط الاختبار.



Example: finding the sum of all integers from 1 to N.

إيجاد مجموع الأعداد الصحيحة من 1 إلى N باستخدام الحلقة for و الحلقة while

```
Algorithm  findSum1
 var  sum, i, n : integer
 begin
   read (n)
    i ← 1
   while (i < = n) do
      sum = sum + i
      i = i + 1
    end while
   write ( sum)
end
```

```
Algorithm  findSum2
 var  sum, i, n : integer
 begin
   read (n)
 for ( i← 1 , n) do
      sum = sum + i
 end for
   write ( sum)
end
```