Fundamentals of Loop in Fortran language

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.

1. Types of programming loops

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

a. DO loop :

We use a **Do** 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

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

The general shape of a complete loop in Fortran is the following :

```
Do var = init_value , final_value  , step_value

  // instructions set  repeated

end do
```

Example : Here is a Fortran program that displays the first 10 integers from 1 to 10 and inversed(from 10 to 1)

```
program numbers
implicit none
integer::i
print*, 'Here are the first 10 numbers:'
do i =1,10
print*, i
end do
end program numbers
```

```
program numbers_invers
implicit none
integer::i
print*, 'Here are the first 10 numbers inversed:'
do i =10,1,-1
print*, i
end do
end program numbers_invers
```

DO While loop in programming

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

تتكون الحلقة **Do 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**
Do while (loop condition)

loop body
condition update

end do

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

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

```
program numbers_invers
implicit none
integer::i,n,s
print*,'Program sum of N first integer numbers:'
 read*,n
 s=0
do i =1,n
 s=s+i
end do
print*, "sum=",s
end program numbers_invers
```

```
program numbers_invers
implicit none
integer::i,n,s
print*,'Program sum of N first int
 read*,n
 s=0
 i =1
do while(i<=n)
 s=s+i
 i=i+1
end do
print*, "sum=",s
end program numbers_invers
```