

Loop Control Statements

You can use repetitive statements when you want to repeat the implementation of one or more statements. There are five different types of repetitive statements that you can use are

for-to-do

for-downto-do

foreach-in-do

while-do

repeat-until

for-to-do

If you use the for statement, you should already know how many times you'll repeat the implementation

```
intCount: Integer;
```

```
Total: Integer
```

```
for intCount:=1 to 5 do
```

```
    total+=incount;
```

for-downto-do

with fordownto statement you can decrement by using fordownto statement, which counts downwards

```
intCount: Integer;
```

```
Total: Integer
```

```
for intCount:=5 downto 1 do
```

```
  total+=incount;
```

foreach

The foreach statement can only be used on Enumerable collection (List & Dictionary),
and can't be used on Arrays

```
EmployeeNames: List of [Text[100]]
```

```
EmployeeName: Text[100]
```

```
foreach EmployeeName in EmployeeNames do
```

```
    Message('EmployeeName: %1', EmployeeName)
```

While

The while condition first checks to see if the condition is true before you start looping. Providing that this condition stays true, it keeps running the statements within your while block

```
while totalSales < 8 do begin  
    index := index + 1;  
    totalSales := totalSales + Sales[index];  
end;
```

Repeat Until

The repeat until statement runs first and then checks for a condition. It loops until the condition is valid, meaning that repeat until statements are run at least once

repeat

until (condition)

The repeat until statement is often used to loop through the records.