

Repetition in Turing

Nested Counted Loops

Accumulating Values

Accumulating Values

In many applications, we want to find the total amount of some quantity.

For example:

- adding student grades
- cash register totals
- scoring a game

Sample Code – Adding 3 Numbers

```
var num, sum : int
sum := 0      % initialize sum to zero

put "This program will add 3 integers"

for count : 1 .. 3
    put "Enter number ", count, ": " ..
    get num
    sum := sum + num
end for

put "The sum is ", sum
```

Example Problem

Write a program that will output the values "10 20 30" on the screen.

Have this pattern repeat 5 times.

Sample Output:

10

20

30

10

20

30

...

Example Problem

Consider a simple FOR loop, which counts from 10 to 30:

```
for j : 10 .. 30 by 10
    % j will be 10, 20, 30
    put j
end for
```

Example Problem

We could duplicate this code multiple times, which would produce the desired effect:

```
for j : 10 .. 30 by 10
  % j will be 10, 20, 30
  put j
end for
```

```
for j : 10 .. 30 by 10
  % j will be 10, 20, 30
  put j
end for
```

```
for j : 10 .. 30 by 10
  % j will be 10, 20, 30
  put j
end for
```


Example Problem

We could duplicate this code multiple times, which would produce the desired effect:

```
for j : 10 .. 30 by 10  
  put j  
end for
```

```
for j : 10 .. 30 by 10  
  put j  
end for
```

```
for j : 10 .. 30 by 10  
  put j  
end for
```



This is just the same code, repeated multiple times.

To repeat identical code, we use loops


Example Problem

We could duplicate this code multiple times, which would produce the desired effect:

```
for j : 10 .. 30 by 10
  put j
end for
```

```
for j : 10 .. 30 by 10
  put j
end for
```

```
for j : 10 .. 30 by 10
  put j
end for
```



```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put j
  end for
end for
```


Example Problem

Our original problem statement wanted to output the sequence "10 20 30" five times, so we need a small adjustment to our solution:

```
for i : 1 .. 5
  for j : 10 .. 30 by 10
    put j
  end for
end for
```

Nested Loops

```
for i : 1 .. 3          i
  for j : 10 .. 30 by 10  j
    put i * j          i * j
  end for
end for
```

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3          i      j      i * j
  for j : 10 .. 30 by 10
    put i * j          1
  end for
end for
```

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

| i | j | i * j |
|---|----|-------|
| 1 | 10 | |

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

| i | j | i * j |
|---|----|-------|
| 1 | 10 | 10 |

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

| i | j | i * j |
|---|----|-------|
| 1 | 10 | 10 |
| 1 | 20 | 20 |

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

| i | j | i * j |
|---|----|-------|
| 1 | 10 | 10 |
| 1 | 20 | 20 |
| 1 | 30 | 30 |

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

| i | j | i * j |
|---|----|-------|
| 1 | 10 | 10 |
| 1 | 20 | 20 |
| 1 | 30 | 30 |
| 2 | 10 | 20 |

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

| i | j | i * j |
|---|----|-------|
| 1 | 10 | 10 |
| 1 | 20 | 20 |
| 1 | 30 | 30 |
| 2 | 10 | 20 |
| 2 | 20 | 40 |

Try a “walkthrough” of the code to determine what it does

Nested Loops

```
for i : 1 .. 3
  for j : 10 .. 30 by 10
    put i * j
  end for
end for
```

Try a “walkthrough” of the code to determine what it does

| i | j | i * j |
|---|----|-------|
| 1 | 10 | 10 |
| 1 | 20 | 20 |
| 1 | 30 | 30 |
| 2 | 10 | 20 |
| 2 | 20 | 40 |
| 2 | 30 | 60 |
| 3 | 10 | 30 |
| 3 | 20 | 60 |
| 3 | 30 | 90 |