Technical Informatics I

Control Structures (Repetition)

`while` and `do-while`

by
Dr. Fatimah
Faculty of Mechanical Engineering
fatimahd@ump.edu.my

Control Structures (Repetition)

- Aims
  - Introduce students to Control Structures (Repetition): `while`
  - Introduce students to Control Structures (Repetition): `do-while`

- Expected Outcomes
  - Students are able to construct simple C programs that can implement repetition control structures `while`
  - Students are able to construct simple C programs that can implement repetition control structures `do-while`

- References
  - Harry H. Cheng, 2010. C for Engineers and Scientists: An Interpretive Approach, McGraw Hill

Content

- Selection Structures: `while`
- Selection Structures: `do-while`
- Nested Loops
  - Nested `while`
  - Nested do-`while`
- Examples
- Conclusion

Control structures

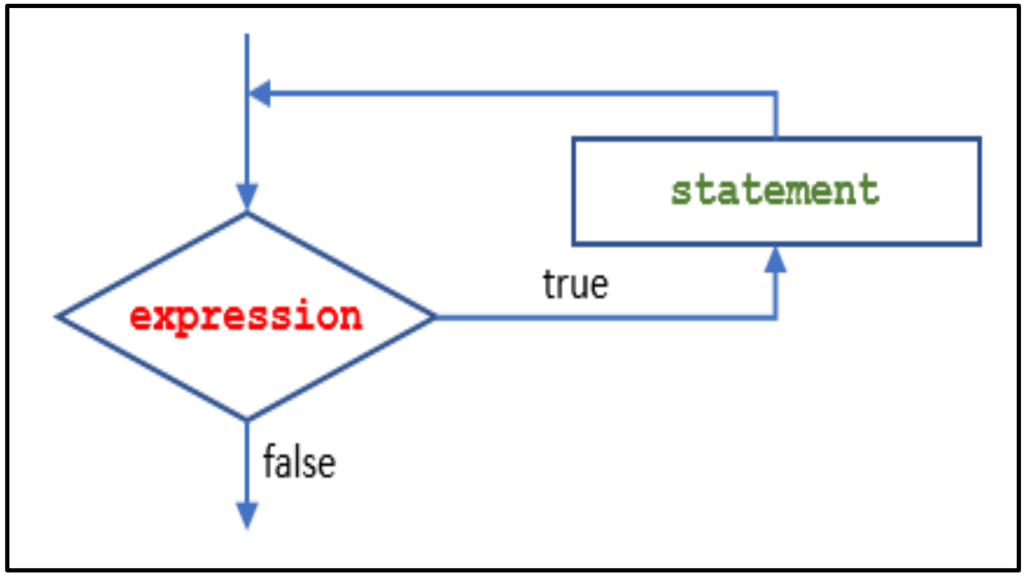
- There are 3 control structures for C programs:
  1. **Sequence**
     - Each statement is executed sequentially (as seen in the previous lectures
  2. **Selection**
     - One statement is *selected* over another depending on a Selection
       – If, else if, else & switch
       – If var1 > 10, do *this*…, else do *that*…
  3. **Repetition**
     - Statements are *repeatedly* executed until it meets a certain *condition*
       – for, while, do-while loops

Control Structure – Repetition (Overview)

- Three kinds of selections structures
  - **`while`** Loop
  - **`do-while`** Loop
  - **`for`** Loop
- Loops Consist of
  - 1: Loop Initialization
  - 2: Terminating Condition (Loop Guard)
  - 3: Loop Body (Statement)
  - 4: Terminating Action

Control Structure (Repetition): `while`

- The flow chart and the syntax of a while loop is given as follows where `statement` will be executed until `expression` is FALSE.



```
while (expression) {
    statement
}
```

Control Structure (Repetition): `while`

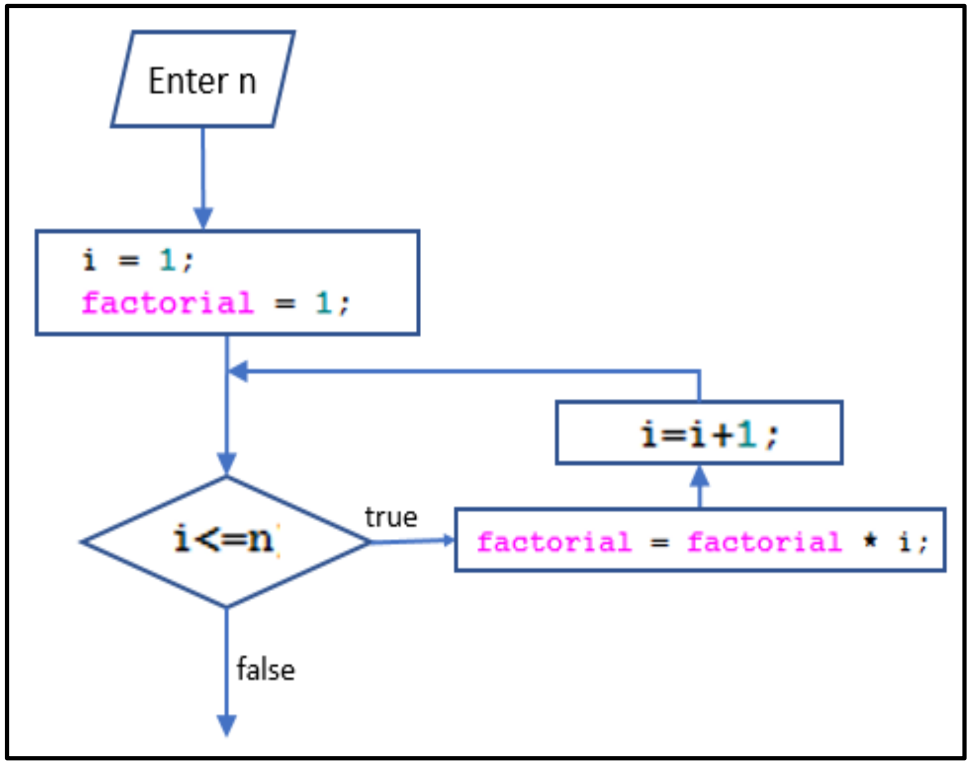
- **Example 1:**

Write a code that calculates the factorial of n where n is the user input using `while` loops

For example: If the use inputs n = 5, then the code will return 120 since

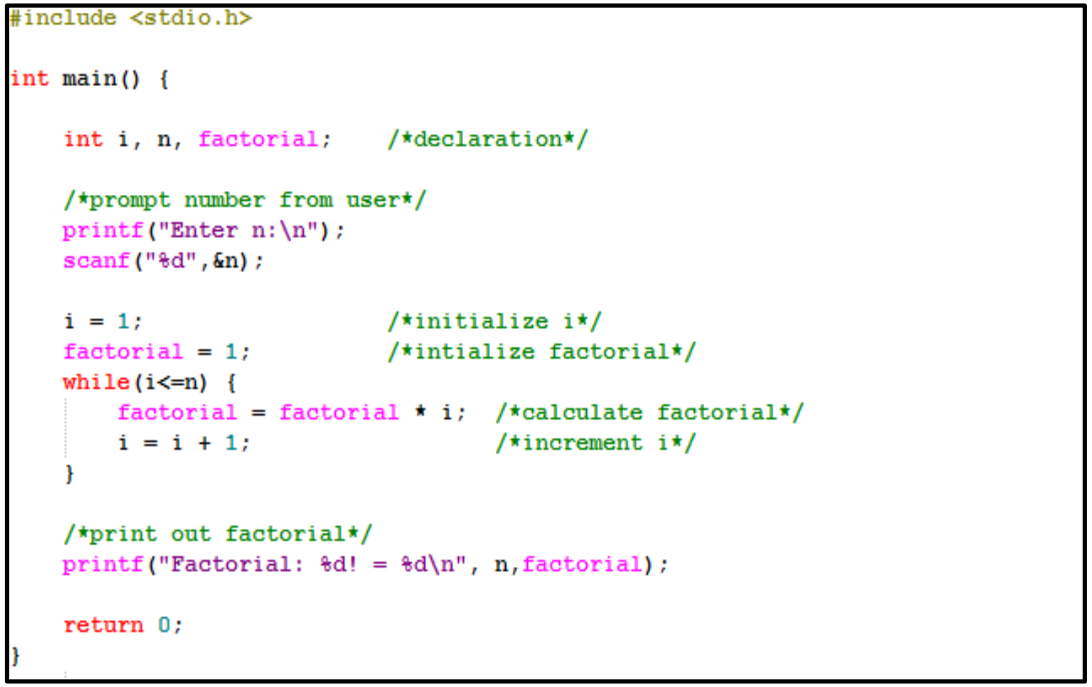
$$5! = 1*2*3*4*5 = 120$$

Control Structure (Repetition): `while`

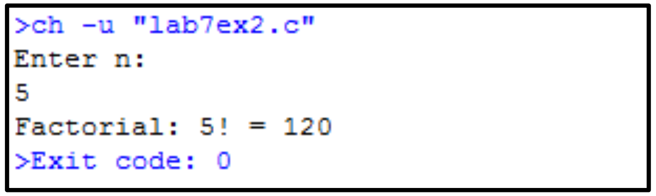


```
scanf("%d",&n);
i = 1;
factorial = 1;
while(i<=n) {
        factorial = factorial * i;
        i = i + 1;
}
```

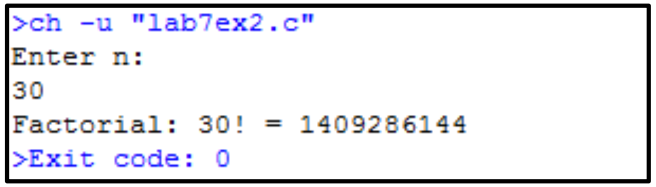
Control Structure (Repetition): `while`



```c
#include <stdio.h>

int main() {

    int i, n, factorial;    /*declaration*/

    /*prompt number from user*/
    printf("Enter n:\n");
    scanf("%d",&n);

    i = 1;                  /*initialize i*/
    factorial = 1;          /*intialize factorial*/
    while(i<=n) {
        factorial = factorial * i;  /*calculate factorial*/
        i = i + 1;                  /*increment i*/
    }

    /*print out factorial*/
    printf("Factorial: %d! = %d\n", n,factorial);

    return 0;
}
```



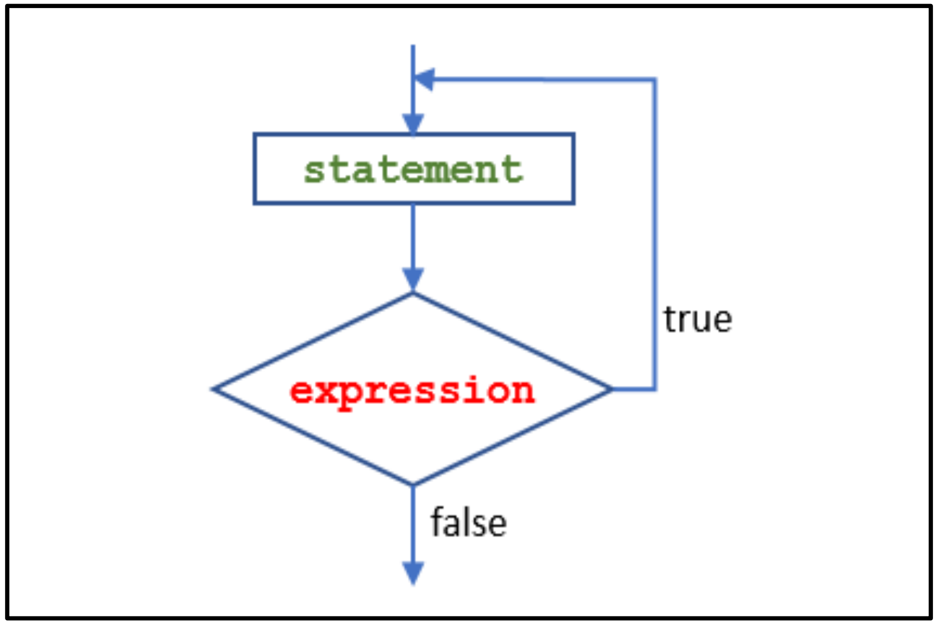
```
>ch -u "lab7ex2.c"
Enter n:
5
Factorial: 5! = 120
>Exit code: 0
```



```
>ch -u "lab7ex2.c"
Enter n:
30
Factorial: 30! = 1409286144
>Exit code: 0
```

Control Structure (Repetition): `do-while`

- The flow chart and the syntax of a `do-while` loop is given as follows where **`statement`** will be executed until **`expression`** is FALSE.



```
do{
    statement
} while (expression);
```

Control Structure (Repetition): `do-while`

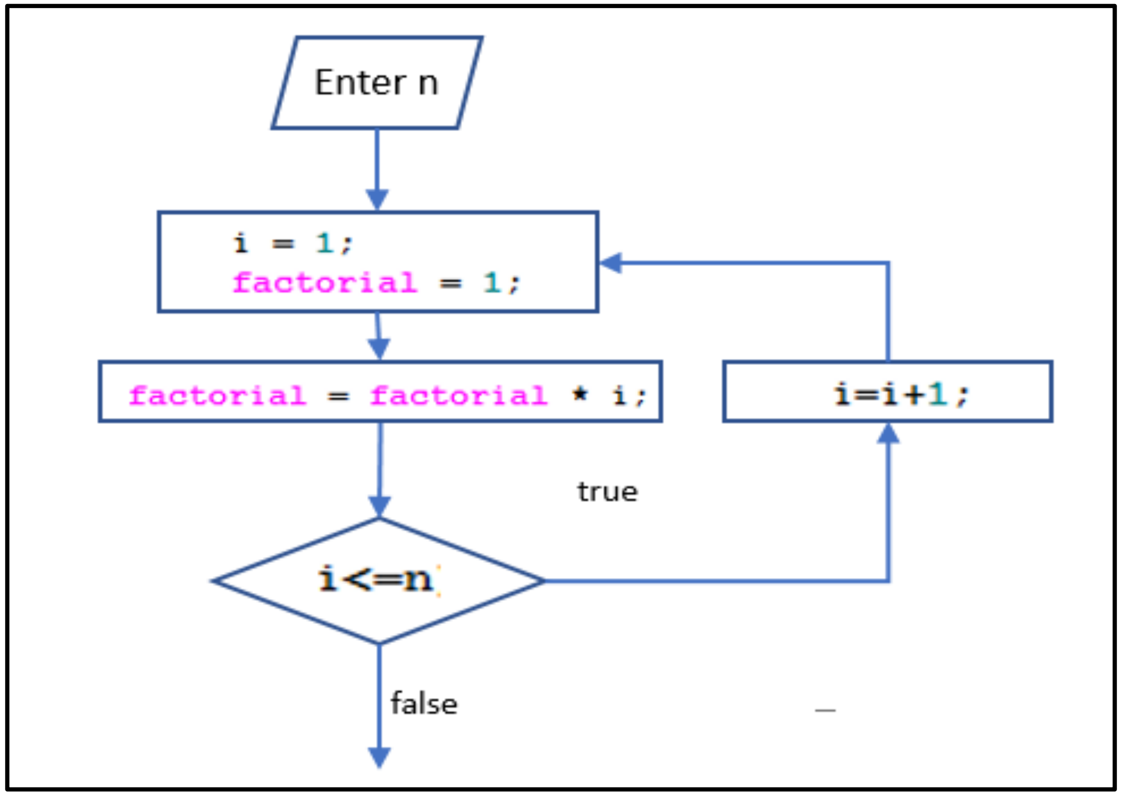
- **Example 2:**

Write a code that calculates the factorial of n where n is the user input using a `do-while` loop

For example: If the use inputs n = 5, then the code will return 120 since

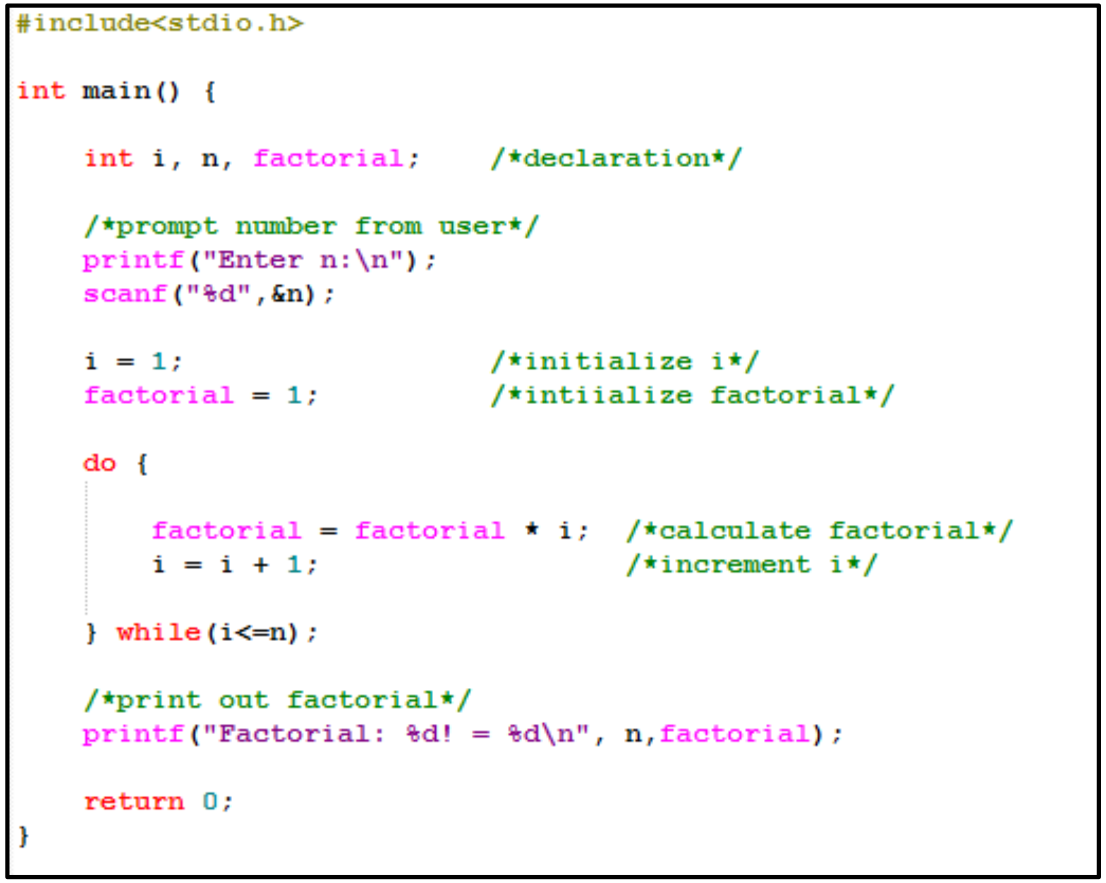
5! = 1*2*3*4*5 = 120

Control Structure (Repetition): `do-while`

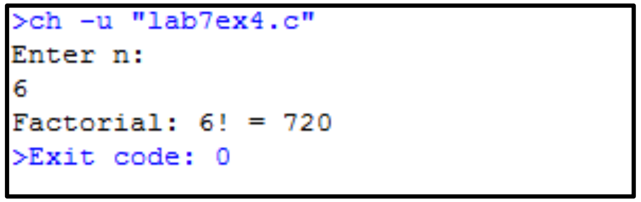


```
scanf("%d",&n);
i = 1;
factorial = 1;
do {

    factorial = factorial * i;
    i = i + 1;

} while(i<=n);
```

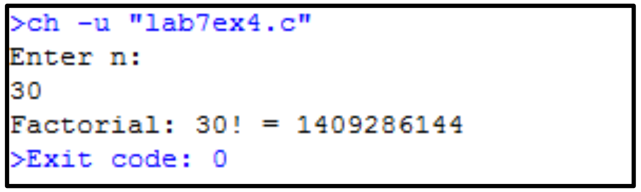
Control Structure (Repetition): `do-while`



```c
#include<stdio.h>

int main() {

    int i, n, factorial;    /*declaration*/

    /*prompt number from user*/
    printf("Enter n:\n");
    scanf("%d",&n);

    i = 1;                  /*initialize i*/
    factorial = 1;          /*intiialize factorial*/

    do {

        factorial = factorial * i;  /*calculate factorial*/
        i = i + 1;                  /*increment i*/

    } while(i<=n);

    /*print out factorial*/
    printf("Factorial: %d! = %d\n", n,factorial);

    return 0;
}
```



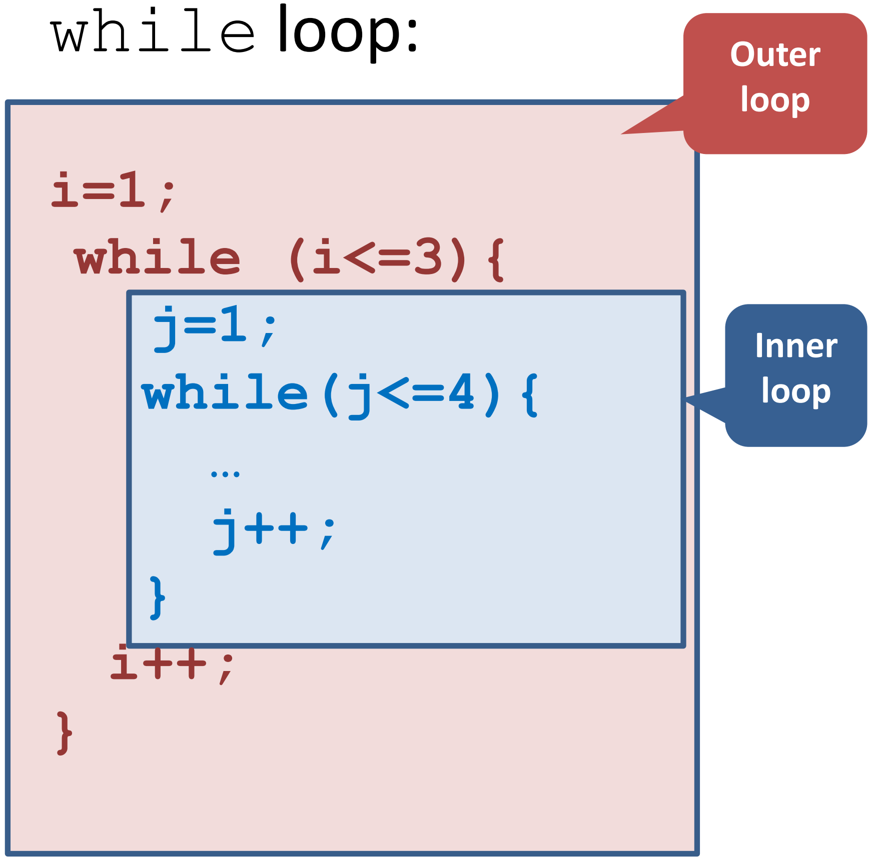
```
>ch -u "lab7ex4.c"
Enter n:
6
Factorial: 6! = 720
>Exit code: 0
```



```
>ch -u "lab7ex4.c"
Enter n:
30
Factorial: 30! = 1409286144
>Exit code: 0
```

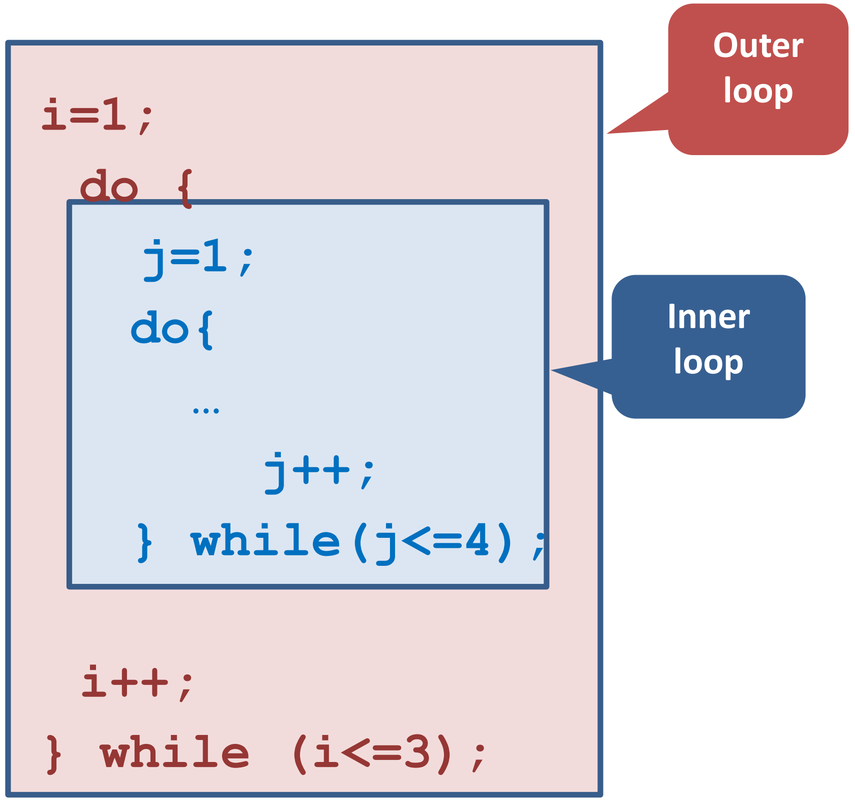
Nested Loops

- `while` loop:



```
i=1;
 while (i<=3){
   j=1;
   while(j<=4){
     …
     j++;
   }
   i++;
}
```

- `do-while` loop:



```
i=1;
 do {
    j=1;
    do{
      …
         j++;
    } while(j<=4);

  i++;
} while (i<=3);
```

- The above loops will run for 3*4 iterations.
- The above loops will run for 3*4 iterations.

Technical Informatics I

Lecture 6

Dr Fatimah