



Mahidol University

ITCS113

Fundamentals of Programming

Lecture 4 - Repetition (cont.)

Instructor: Asst. Prof. Dr. Akara Supratak

Contact: akara.sup@mahidol.edu

Common Mistakes in Quiz#1

- Forget to save the file
- Did not open “Desktop” folder
 - Your terminal will not see the C file that you have written
- Did not know how to compile and run the code

Agenda

- `do-while` statement (or `do-while-loop`)
 - Input data validation
- `while/for/do-while` loops
- Altering the flow of loops with `break` and `continue`



do-while **statement**

Type of Loop

- Pre-test loops

- Evaluate an expression at the **start** of the repetition
for, while

- Post-test loops

- Evaluate an expression at the **end** of the repetition, i.e.,
Your loop should execute **at least one time**
do-while
- Use for input validation: need to receive an input before validation

do-while statement

- Typically, used when we want the statements within the loop to be executed **at least once**.

do-while statement (flowchart)

Example

- What is the objective of this program?

```
#include <stdio.h>

int main() {
    int n;
    do {
        printf("Enter a non-negative number: ");
        scanf("%d", &n);
    } while(n < 0);

    printf("Input number: %d\n", n);
    return 0;
}
```




while **vs.** do-while

Example: while Input validation

Input validation: input is only single digit

```
int counter=0; // For counting odd numbers
char n;        // For user input
scanf(" %c", &n);

// Use while for number selection and validation
while(n != 'q'){

    if(n>='0' && n<='9'){
        int num = n - '0';    // Convert ASCII to number

        if(num%2 == 1){    // Check if ODD number
            counter++;
        }
    }
    scanf(" %c", &n);
}
printf("%d", counter);
```

Example: do-while Input validation

Input must be only single digits

```
int counter=0; // For counting odd numbers
char n;        // For user input

// Use while for number selection and validation
do {
    scanf(" %c", &n);
    if(n>='0' && n<='9'){
        int num = n - '0';    // Convert ASCII to number

        if(num%2 == 1){    // Check if ODD number
            counter++;
        }
    }
} while(n != 'q');

printf("%d", counter);
```

Example: do-while **vs.** while



while/for/do-while

while/for/do-while

Step-by-step in the loop design



Altering the flow of loops with `break` **and** `continue`

Altering the flow of loops

`break; statement`

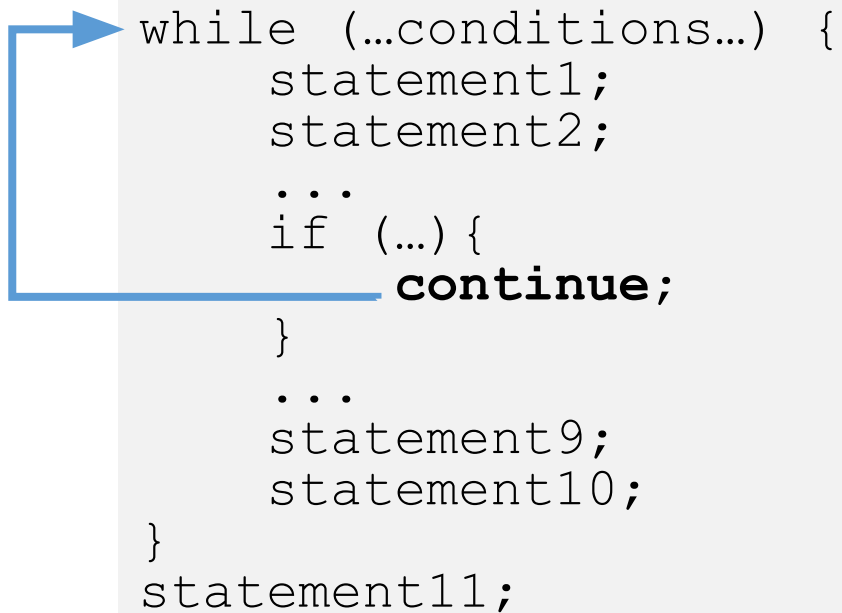
Example

Write a program to receive and print input numbers until a user input a *negative number*

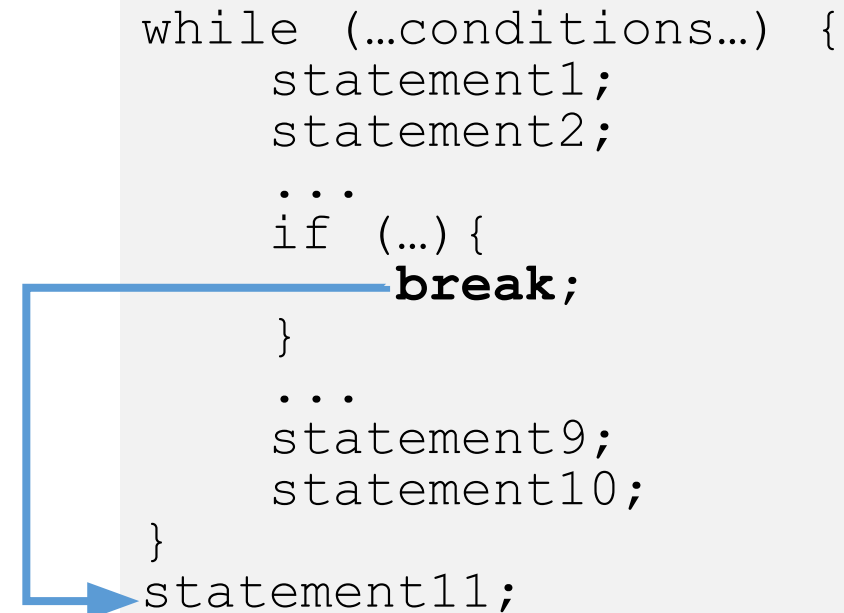
```
#include <stdio.h>
int main() {
    // Initialize statement
    int x = 1;
    int n;
    while (x) // Expression or Condition
    {
        // Statements
        scanf("%d", &n);
        printf("%d\n", n);
        // Alteration
        if (n < 0)
            x=0;
    }
    return 0;
}
```

`continue; statement`

break; VS. continue;



```
while (...conditions...) {  
    statement1;  
    statement2;  
    ...  
    if (...) {  
        continue;  
    }  
    ...  
    statement9;  
    statement10;  
}  
statement11;
```



```
while (...conditions...) {  
    statement1;  
    statement2;  
    ...  
    if (...) {  
        break;  
    }  
    ...  
    statement9;  
    statement10;  
}  
statement11;
```

Exercise

- What is the output of the following program?

```
#include <stdio.h>
int main()
{
    int counter = 10;
    while(counter >= 0) {
        if (counter == 7) {
            continue;
        }
        counter--;
        printf("%d\n", counter);
    }
    return 0;
}
```



Lab Exercises