

CS1100 – Introduction to Programming

- Programming : From Turtle to C.
- Data Types in C, Representations, Operators.
- Formatting the Input and the Output.
- Execution of Programs, Compilers.
- Modifying the control flow in Programs
`if-then-else`, `switch`.

} Week 1-4

CS1100 – Introduction to Programming

- Programming : From Turtle to C.
- Data Types in C, Representations, Operators.
- Formatting the Input and the Output.
- Execution of Programs, Compilers.
- Modifying the control flow in Programs
if-then-else, switch.

Week 1-4

-
- while, for, do while constructs in C.
 - Example problems.
 - Programming for engineers.

Week 5-6

Multiple Selection : **switch** construct

- Syntax:

```
switch (expression) {  
    case const-expr-1: statements-1;  
    case const-expr-2: statements-2;  
    ⋮  
    default: statements-def;  
}
```

- Semantics

- Evaluate expression.
- If it matches the const-expr-i, execute statements-i and following as long as an **explicit break** is not encountered.
- If none of the const-expr-i match, execute the statements-def.

Even or odd – using switch

Given a number determine if it is even or odd.

```
#include<stdio.h>
int main() {

    int input;

    printf("Enter an integer: \t");
    scanf("%d", &input);

    switch (input % 2) {
        case 0: printf("Number is even\n");
        case 1: printf("Number is odd\n");
        default: printf("Neither even nor odd\n");
    }
    return 0;
}
```

Even or odd – using switch

Given a number determine if it is even or odd.

```
#include<stdio.h>
int main() {

    int input;

    printf("Enter an integer: \t");
    scanf("%d", &input);

    switch (input % 2) {
        case 0: printf("Number is even\n");
        case 1: printf("Number is odd\n");
        default: printf("Neither even nor odd\n");
    }
    return 0;
}
```

- What is the output of the program on 10, 5, 13?
- Observe the missing break.

Even or odd – using switch – a correct program

Given a number determine if it is even or odd.

```
#include<stdio.h>
int main() {

    int input;

    printf("Enter an integer: \t");
    scanf("%d", &input);

    switch (input % 2) {
        case 0: printf("Number is even\n"); break;
        case 1: printf("Number is odd\n"); break;
        default: printf("Neither even nor odd\n");
    }
    return 0;
}
```

The **while** construct

- Syntax

```
while (expression) {  
    statements;  
}
```

The **while** construct

- Syntax

```
while (expression) {  
    statements;  
}
```

- Semantics

1. As long as expression is true, execute statements.
2. If expression is false, exit the loop.

The **while** construct

- Syntax

```
while (expression) {  
    statements;  
}
```

- Semantics

1. As long as expression is true, execute statements.
 2. If expression is false, exit the loop.
- Value of expression must be changed by the body of the loop, otherwise we have an infinite loop.

The **while** construct

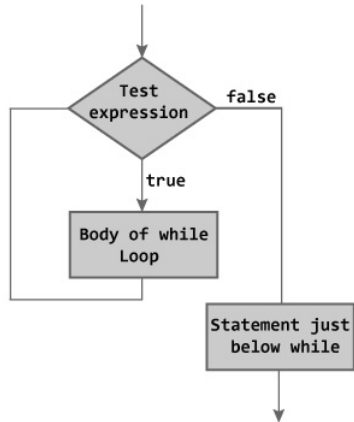
- **Syntax**

```
while (expression) {  
    statements;  
}
```

- **Semantics**

1. As long as expression is true, execute statements.
 2. If expression is false, exit the loop.
- Value of expression must be changed by the body of the loop, otherwise we have an infinite loop.
 - expression can contain relational, logical or equality operators.

Relational	<=	<	>	>=
Equality	==	!=		
Logical	&&			



Reversing the digits of a given unsigned integer

```
#include "stdio.h"
int main () {
    int number, revNumber, remainder;
    revNumber = 0;
    printf ("Input number:");
    scanf ("%d", &number);
    while (number > 0) {
        remainder = number % 10;
        revNumber = revNumber*10 + remainder;
        number = number/10;
    }
    printf ("The reversed number is : %d\n", revNumber);
}
```

Example: Sum even and odd numbers

Accept integers from the standard input as long as the user does not enter -1. Once the user enters -1, print the sum of all integers entered so far, sum of even integers and sum of odd integers.

Two useful constructs:

- while loop repetitive statement
- switch multiple selection

Summing up odd and even numbers

Is the program correct?

```
#include<stdio.h>
int main() {
    int input;
    int sum, eSum, oSum;
    printf("Enter an integer: \t");
    scanf(" %d", &input);

    while (input != -1) {
        sum += input;
        switch (input % 2) {
            case 0: eSum += input; break;
            case 1: oSum += input;
        }
    }
    printf("sum = %d, oddSum = %d, evenSum = %d\n", sum, oSum, eSum);
    return 0;
}
```

Summing up odd and even numbers

Is the program correct?

```
#include<stdio.h>
int main() {
    int input;
    int sum, eSum, oSum;
    printf("Enter an integer: \t");
    scanf(" %d", &input);

    while (input != -1) {
        sum += input;
        switch (input % 2) {
            case 0: eSum += input; break;
            case 1: oSum += input;
        }
    }
    printf("sum = %d, oddSum = %d, evenSum = %d\n", sum, oSum, eSum);
    return 0;
}
```

- common mistake: forgotten initialization.
- expr. not modified in body of loop.

Summing up odd and even numbers

```
#include<stdio.h>
int main() {
    int input;
    int sum, eSum, oSum;
    printf("Enter an integer: \t");
    scanf(" %d", &input);
    sum = eSum = oSum = 0; // initialization.

    while (input != -1) {
        sum += input;
        switch (input % 2) {
            case 0: eSum += input; break;
            case 1: oSum += input;
        }
        printf("Enter an integer: \t");
        scanf(" %d", &input);
    }
    printf("sum = %d, oddSum = %d, evenSum = %d\n", sum, oSum, eSum);
    return 0;
}
```

for construct

- Syntax

```
for (stmt1; expr; stmt2) {  
    statements;  
}
```

for construct

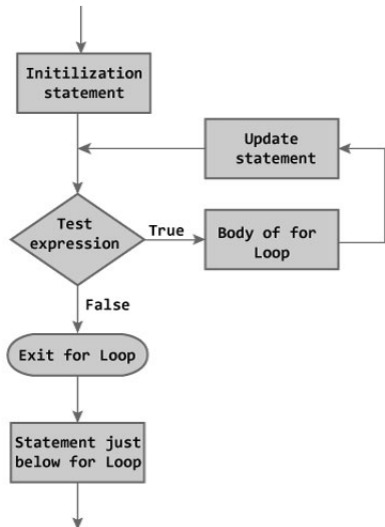
- Syntax

```
for (stmt1; expr; stmt2) {  
    statements;  
}
```

- Semantics

1. Execute stmt1 (**Init.**).
2. If expr is true, execute statements.
execute stmt2 (**Update**).
goto step 2.
3. If expr is false, exit loop.

- **expr must be changed to ensure that it is not an infinite loop.**



Sum of 100 integers – using for loop

User enters 100 integers. Sum them and print the sum.

Sum of 100 integers – using for loop

User enters 100 integers. Sum them and print the sum.

```
#include<stdio.h>
int main() {
    int input;
    int count;
    int sum = 0;

    for (count=1; count <= 100; count++) {
        printf("Enter an integer: \t");
        scanf(" %d", &input);
        sum += input;
    }
    printf("sum of numbers entered is %d\n", sum);
    return 0;
}
```

Typically, stmt1 : initialization and stmt2 : increment.