



CAIRO UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

LANGUAGES AND COMPILERS

Project Phase 1

Team 7

Remonda Talaat Eskarous

SEC:1, BN:19

Mohamed Shawky Zaky

SEC:2, BN:15

Mohamed Ahmed Mohamed Ahmed

SEC:2, BN:10

Ahmed Mohamed Zakaria

SEC:1, BN:3

1 Syntax

The overall syntax follows the C/C++ syntax, with some simplifications and modifications.

1.1 Variables and Constants Declaration

- Variables are declared using :
 - *DATA_TYPE IDENTIFIER = VALUE;*
 - *DATA_TYPE IDENTIFIER;*
- Constants are declared using :
 - *const DATA_TYPE IDENTIFIER = VALUE;*
- Identifiers can contain uppercase or lowercase characters, as well as numbers and underscores. However, they must begin with a character.
- Data types include *int*, *float*, *char*, *string* and *bool*.
- *string* data type can only have **letters**, **digits** and **spaces**.

1.2 Mathematical and Logical Expressions

- Mathematical operators : +, −, *, /, %
- Logical operators : &&, ||, !, along with comparators.

1.3 Assignment Statement

The assignment statement is done through =, whether it's an initialization (as mentioned above) or change of value. However, we **don't** include ++ and -- operators, nor +=, -= ..., etc.

1.4 If-Then-Else Statements

The *if* statements are written in the same way as in C/C++ and the block structures are defined using {}. For example :

```
if (CONDITION)

{ STATEMENT; }

else

{ STATEMENT; }
```

1.5 While Loops

The *while* statements are written in the same way as in **C/C++** and the block structures are defined using `{}`. For example :

```
while (CONDITION) {  
  
    STATEMENT;  
  
}
```

1.6 Do-While Loops

The *do-while* statements are written in the same way as in **C/C++** and the block structures are defined using `{}`. For example :

```
do {  
  
    STATEMENT;  
  
} while (CONDITION);
```

1.7 For Loops

The *for* statements are written in the same way as in **C/C++** and the block structures are defined using `{}`. For example :

```
for (IDENTIFIER = VALUE; IDENTIFIER < VALUE; IDENTIFIER = IDENTIFIER + 1) {  
  
    STATEMENT;  
  
}
```

NOTE : There can be any kind of expressions between the *semicolons* of *for* statements.

1.8 Switch Statements

The *switch* statements are written in the same way as in **C/C++** and the block structures are defined using `{}`. For example :

```
switch (IDENTIFIER) {  
  
    case VALUE:  
  
        STATEMENT;  
  
    break;
```

.

.

.

default:

STATEMENT;

}

NOTE : The *switch* statement can have only one case or *default* without any cases.

1.9 Functions

The *functions* are written in the same way as in **C/C++** and the block structures are defined using {}.

The function can have a **return value** :

DATA_TYPE IDENTIFIER(DATA_TYPE IDENTIFIER, ...) {

STATEMENT;

RETURN IDENTIFIER (or VALUE);

}

Also, the function can have **no return value** :

void IDENTIFIER(DATA_TYPE IDENTIFIER, ...) {

STATEMENT;

}

NOTE : In order to call a function, it must be defined above the call site. Also, there *cannot* exist a function prototype without definition.