**Introduction to programming with C**

**DO NOT USE HELP FROM INTERNET FOR THIS EXERSCISE!!!DO IT BY YOURSELF!!**

Structure of a C program

|  |  |
| --- | --- |
| Header | #include <stdio.h> |
| main() | int main()  { |
| Variable declaration | int a = 10; |
| Body | printf(“%d”,a); |
| Return | return 0;  } |

The components of the above structure are:

**Header Files Inclusion**: The first and foremost component is the inclusion of the Header files in a C program.

A header file is a file with extension .h which contains C function declarations and macro definitions to be shared between several source files.

Some of C Header files:

stddef.h – Defines several useful types and macros.

stdint.h – Defines exact width integer types.

stdio.h – Defines core input and output functions

stdlib.h – Defines numeric conversion functions, pseudo-random network generator, memory allocation

string.h – Defines string handling functions

math.h – Defines common mathematical functions

Syntax to include a header file in C:

**#include <(header\_file\_name).h>**

**Main Method Declaration**: The next part of a C program is to declare the main() function. The syntax to declare the main function is:

Syntax to Declare main method:

**int main()**

**{}**

**Variable Declaration**: The next part of any C program is the variable declaration. It refers to the variables that are to be used in the function. Please note that in C program, no variable can be used without being declared. Also in a C program, the variables are to be declared before any operation in the function.

int main()

{

int a;

.

.

**Body**: Body of a function in C program, refers to the operations that are performed in the functions. It can be anything like manipulations, searching, sorting, printing, etc. Note the semicolon (;). It is used to indicate end of statement.

int main()

{

int a;

printf("%d", a);

.

.

**Return Statement**: The last part in any C program is the return statement. The return statement refers to the returning of the values from a function. This return statement and return value depend upon the return type of the function. For example, if the return type is void, then there will be no return statement. In any other case, there will be a return statement and the return value will be of the type of the specified return type.

int main()

{

int a;

printf("%d", a);

return 0;

}

**Comments**

Comments are like helping text in your C program and they are ignored by the compiler. They start with /\* and terminate with the characters \*/ as shown below −

**/\* my first program in C \*/**

You cannot have comments within comments and they do not occur within a string or character literals.

**Escape Sequences**

Sometimes, it is necessary to use characters that cannot be typed or has special meaning in C programming. For example: newline(enter), tab, question mark etc.

\n Newline

\t Horizontal tab

**Variables**

In programming, a variable is a container (storage area) to hold data.

To indicate the storage area, each variable should be given a unique name (identifier). Variable names are just the symbolic representation of a memory location. For example:

**int playerScore = 95;**

Here, playerScore is a variable of int type. Here, the variable is assigned an integer value 95.

The value of a variable can be changed, hence the name variable.

**Rules for naming a variable**

A variable name can have only letters (both uppercase and lowercase letters), digits and underscore.

The first letter of a variable should be either a letter or an underscore.

There is no rule on how long a variable name (identifier) can be. However, you may run into problems in some compilers if the variable name is longer than 31 characters.

Note: You should always try to give meaningful names to variables. For example: firstName is a better variable name than fn.

C is a strongly typed language. This means that the variable type cannot be changed once it is declared.

**C Data Types**

In C programming, data types are declarations for variables. This determines the type and size of data associated with variables. For example,

int myVar;

Here, myVar is a variable of int (integer) type. The size of int is 4 bytes.

**Some Basic types**

Here's a table containing commonly used types in C programming:

Type Size (bytes) Format Specifier

int at least 2, usually 4 %d

char 1 %c

float 4 %f

double 8 %lf

**C library function - printf()**

The C library function int printf(const char \*format, ...) sends formatted output to stdout.

**int printf(const char \*format, ...)**

format − This is the string that contains the text to be written to stdout

Specifier & Output

**c** -Character

**d** or **i** - Signed decimal integer

**int a = 10;**

**printf(“%d”,a);**

For the exercises use the online compiler and debugger <https://www.onlinegdb.com/>.

Select C Language in the upper right corner selection list.

**Exercise 1.** Open the onlinegdb compiler. Observe the default program.

1. Find each part of the structure of a C program in it. Mark it with a comment.

2. Compile and run the program. See the result in the console.

3. Change the program to print “Hello” with your name.

4. Change the program to print(with the new lines):

Hello, <your name>!

How are you?

What are you doing today?

**Exercise 2.** Declare a variable of type int and initialize it with a value.

1. Print the value.
2. \*Write 3 good variable names and 3 bad variable names
3. Declare another variable and initialize it with another value.
4. Print both values.
5. Print the sum of the values.