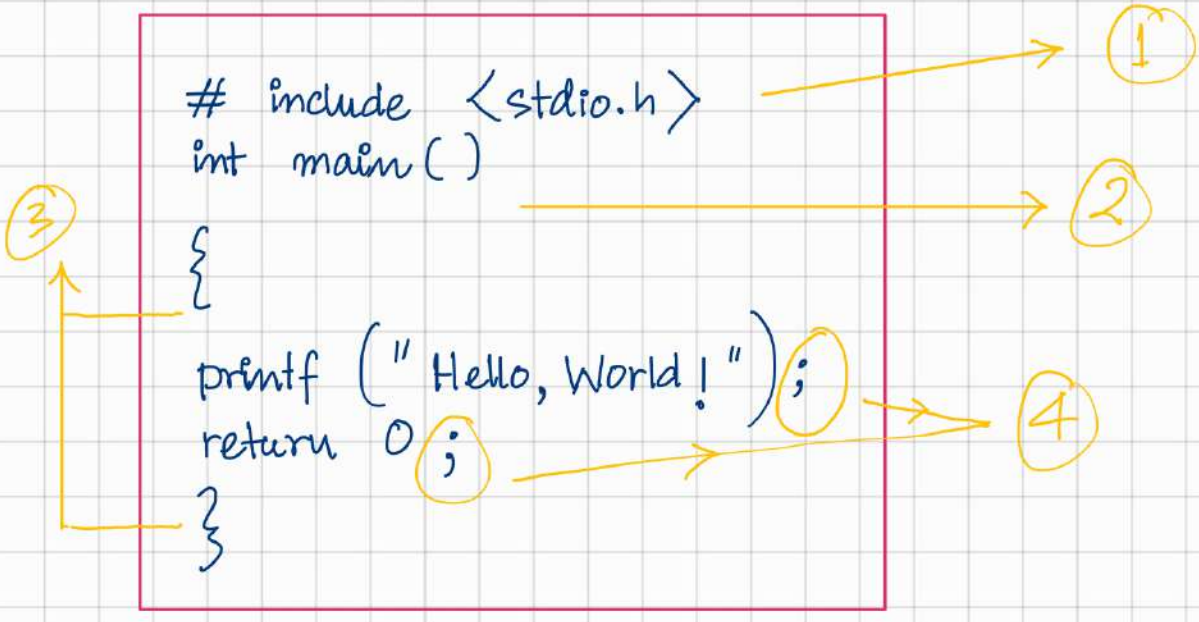
"Hello World" program

↳ your first program in C.



1. `#include <stdio.h>`

this is a preprocessor command that includes standard input / output header file - "stdio.h" from C-library before compiling a C-program.

Another frequently used preprocessor command is -

`#include <math.h>`

this is a preprocessor command that includes standard mathematic header file - "math.h" from C-library function.

(2) int main()

↳ int main() means that our function needs to return some integer at the end of our execution and we do so by returning 0 at the end of the program. 0 is the standard for the "successful execution of the of the program". Always make sure to return 0 at the end of the main().

However, for simple programs we may choose not to specify the datatype of the return value by omiting the 'int' and simply choose to write main(). We shall not write return 0 at the end.

```
# include <stdio.h>
main()
{
  printf("Hello, world!");
}
```

(3) { } any function must be written within curly braces.

(4) ; all C-statements must end with a semicolon.

Commenting in C

1. //--- single line comment line
2. */ --- /* anything written inside it becomes a comment.

---

printf() function in C → to display output

scanf() function in C → to scan/read/input values.

Datatypes in C

int (%d) → integer (1, 4, -6, 0, etc)

float (%f) → floating points (-1.00, 4.13, etc)

(H/W : find out what exactly are floating points and how are they stored in a computer)

str (%s) → string ("abc", "Hello")

Example : Write a program to add two real numbers.

```
# include <stdio.h>

int main ( )

{
        float  a, b, c ;
        scanf ( " Input a  %f", &a) ;
        scanf  ("Input b %f" , &b) ;
        c = a + b
        printf ( " result is  %f", c) ;
        return  0 ;

}
```

C constants, variables and Keywords

A character denotes any alphabet, digit or special symbol used to represent info.

alphabets	→	a, b, c, ... , z
digits	→	0, 1, 2, ... , 9
symbols	→	~, , , @, !, &, ...

Alphabets, digits and special symbols when properly combined forms constants, variables and keywords.

* A constant is an entity that doesnot change.
* A variable is an entity that may change.
* A keyword is a word that carries special meaning.

Types of C- constants
— two main categories

primary constants

- integer constant
- real constant
- character constant

secondary constants

- array
- pointer
- structure
- union
- enum

Rules for constructing integer constant :-

a) an integer constant must have atleast one digit
b) must not have a decimal point.
c) either (+)ve or (−)ve.
d) if no sign precedes, it is assumed (+)ve
e) no commas or blanks allowed within integer constant.
f) allowable range

$$[-2147483648, +2147483648]$$

for gcc compiler.

* range of integer const. depends on compiler.

Rules for constructing real constant :-
(floating point constant)

a) real constant must have at least one digit

b) can be of two forms – (i) fractional form
(ii) exponential form

Fractional form – (eg. 0.00342)
- must have a decimal point
- either (+)ve or (−)ve
- no commas or blanks allowed.

Exponential form – (eg. 3.42E−4)
- part appearing before E → mantissa
- part appearing after E → exponent
- mantissa and exponent part should be separated by e or E.
- mantissa part may have a (+)ve or (−)ve sign
- default mantissa part is (+)ve
- exponent must have atleast one digit which must be a (+)ve or a (−)ve integer.

Rules for constructing character constant

a) a character constant is a single alphabet, single digit or a single special symbol enclosed within single inverted commas.

eg: 'c' , '1'

b) both inverted commas should be left pinting.

Types of C-variables

a) a variable name is any combination of 1 to 31 alphabets, digits or underscores.

eg : sum | Pi | x_1

b) the first character in the variable name can not be a digit. It must be an alphabet or underscore.

eg: x_1 ✓ | 1x ✗

c) no commas and blanks are allowed within a variable name.

eg : apple,one ✗ | apple one ✗

d) no special symbols other than an underscore can be used in variable names.

eg : sum%1 ✗ | sum&1 ✗ | sum_1 ✓

C - Keywords

C-keywords are special combinations of alphabets the are reserved for a particular purpose.

Keywords can not be used as variable names because if we do so, we are trying to assign a new meaning to the keyword which is not allowed.

32 keywords are available in C -

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Form of C-program

a) Each instruction is written as a separate statement
b) Statements must appear in same order in which we wish to execute them.
c) all statements should preferably in small case letters.
d) blank spaces may be inserted between two words to improve the readability of the statement
e) no restriction on position – free form language
f) all statements must end in ;

Operators :

1. Arithmetic Operator :

+	−	*	/

2. Assignment Operator :

=	+=	−=	*=	/=	&=

3. Relational Operator :

>	<	>=	==	!=

4. Logical Operator :

&&	||	!
AND	OR	NOT

5. Conditional Syntax : (condition ? true value : false value)

eg: (a > 100 ? 0 : 1)

6. Increament / decreament operator

pre increament	++i
post increament	i++
pre decreament	--i
post decreament	i--

Classroom Exercise : Write C-programs for the following -

(i) add two numbers a and b.

(ii) area of a circle of radius r

(iii) area of a triangle given its three side a, b, c.

$$\left[s = \frac{a+b+c}{2}, \quad a = \sqrt{s(s-a)(s-b)(s-c)} \right]$$

(iv) area of rectangle

(v) conversion of degree into fahrenheit

$$\left[F = \frac{9}{5}C + 32 \right]$$