Variables & Data Types

10) **Variables in C**

→ Variables are used to access the data & modify the data in the main memory

→ They are labels of memory addresses.

Eg:
```
int x = 20;
int y = 10;
x = 10;
```

→ Give variables a meaningful name.

11) **Variable Naming Rules**

→ Allowed characters → a → z
A → Z
0 → 9
(_)

→ Cannot begin with a digit.

→ Cannot be a Keyword.

Convention

(i) Camel Case (name, currentYear, fullName)

(ii) Snake case (name, current_year, player_name).

(iii) Constants : MAX_AGE, PI, LIMIT.

12) **Datatypes in C**

→ For diff datatypes we need diff format specifiers

Primitive (Fundamental) Types:

(i) Integer types → short, int, long, longlong, ~~una~~
unsigned short, unsigned int, unsigned long

(ii) char types → char, unsigned char (Holds single character)

(iii) Floating types → float, double, long double.

(iv) Others → bool, void. #include <stdbool.h>

Non primitive datatypes

(i) Arrays
(ii) Pointers
(iii) User-defined (struct)

13) Ranges of datatypes in C

→ Overflow is common problem which is addressed here.

Datatype	Bits	Range
char	8	-2^7 to 2^7-1
Unsigned char	8	0 to 2^8-1
short	16	-2^{15} to $2^{15}-1$
int	32	-2^{31} to $2^{31}-1$
Unsigned int	32	0 to $2^{32}-1$
long long	64	-2^{63} to $2^{63}-1$
Unsigned long long	64	0 to $2^{64}-1$
float	32	$\pm 2^{-126}$ to $\pm 2^{128}$
double	64	$\pm 2^{-1022}$ to $\pm 2^{1024}$
bool	8	0 to 1

14) Operator sizeof() in C

→ Returns the no of bytes for a datatype
→ Used with variables / literals.
→ Evaluated on compile time
→ It is an operator & not a func

Syntax
sizeof (datatype);

→ If operations are performed in the sizeof(), then values printed, they will not be shown

15) Global & local variables & their scope

→ Local variables are created within a func

→ Global variables are created outside the func. It is accessible everywhere.

Eg:

```
#include <stdio.h>
int main(){
    int x;
printf("%d", x);
    return 0;
}
```

```
#include <stdio.h> int x=10;
int main(){
printf("%d", x);
    return 0;
}
```

→ Non-initialised global variables return value (0) when called.

→ If we want to use the variable first & then initiased, then we used Keyword (extern.) thus helps in global variables of diff files.

Eg:

```
#include <stdio.h>
extern int x;   // external link
int main(){
printf("%d", x);
    return 0;
}
int x=10;
```

Pair of curly Braces creats a scope

→ If you have local variable & global variable of same name, then while accessing, we will access the "local variable".

→ When we have same variables with diff scopes, then they accessed according to the priority of "inner scope to Global scope".

16) Const in C

→ It fixes a value of (x) & it cannot be modified.
→ It will be only in readOnly format.

Syntax

const datatype variableName = value ;

→ Using pointers we can modify the value.
→ Instead of 'const' we can do #define for a permanent value
#difine is macro.

17) Static Variables in C

→ They are special variables
→ Uninitialised static variable's default value is (0).
→ Static local variables are initialized only once & stay through out the program's life.
→ Static global variables have internal link.

18) Type conversion in C

→ Some of these conversion cause loss of information.

Rules

bool → char → short int → int → unsigned int → long → long long → float → double → long double

→ char letters are converted to their ASCII values & then operations are performed.