C Keywords : C Programming Challenge

Goal : By the end of 21 days, you will be able to:

- Write C programs confidently
- Understand loops, arrays, functions, pointers
- Solve beginner and intermediate programming problems
- Build mini console applications

C Programming Keywords

In C programming, **keywords are reserved words that have a special meaning to the C compiler**.

You **cannot use a keyword as a variable name, function name, or other identifier**.

For example: `int age;`

Here:

- `int` → **keyword**
- `age` → **identifier**
- `;` → **semicolon**

How Many Keywords Are There in C?

For the commonly taught **ANSI C (C90)** standard, there are **32 keywords**.

Here is the complete list:

Category	Keywords
Data Types	`char`, `int`, `float`, `double`, `void`
Decision Making	`if`, `else`, `switch`, `case`, `default`
Loops	`for`, `while`, `do`
Jump Statements	`break`, `continue`, `goto`, `return`
Storage Classes	`auto`, `extern`, `static`, `register`
Type Modifiers	`short`, `long`, `signed`, `unsigned`
User-defined Data Types	`struct`, `union`, `enum`, `typedef`
Other	`const`, `volatile`

All 32 keywords

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

Important: `sizeof` is also a C keyword in this traditional 32-keyword list.

Let's Understand Them Step by Step

1. Data Type Keywords

These tell C **what type of data** a variable can store.

`int`

Stores integers.

```
int age = 20;
```

`float`

Stores decimal values.

```
float price = 25.50;
```

`double`

Stores larger/more precise decimal values.

```
double salary = 50000.75;
```

`char`

Stores a single character.

```
char grade = 'A';
```

`void`

Means **no value / no return type** in appropriate contexts.

```
void display()
{
    printf("Hello");
}
```

2. Decision-Making Keywords

These are used to make decisions.

```
if
if(age >= 18)
{
    printf("Eligible");
}
else
if(age >= 18)
    printf("Eligible");
else
    printf("Not Eligible");
switch
```

Used to select one option from multiple choices.

```
switch(choice)
{
    case 1:
        printf("Light");
        break;

    case 2:
        printf("Fan");
        break;
}
case
```

Represents an option inside `switch`.

```
default
```

Executes when no `case` matches.

3. Loop Keywords

Used for repetition.

`for`

```
for(int i = 1; i <= 5; i++)
{
    printf("%d ", i);
}
```

`while`

```
while(i <= 5)
{
    printf("%d ", i);
    i++;
}
```

`do`

Used with `while` to create a `do-while` loop.

```
do
{
    printf("%d ", i);
    i++;
}
while(i <= 5);
```

4. Jump Keywords

These change the normal flow of execution.

`break`

Stops a loop or `switch`.

```
break;
```

Remember your switchboard example:

```
switch
  ↓
case
  ↓
statement
  ↓
break
  ↓
exit switch
```

`continue`

Skips the current loop iteration and continues with the next iteration.

```
continue;
```

```
return
```

Returns control from a function.

```
return 0;
```

```
goto
```

Transfers control to a labeled statement.

```
goto start;
```

It should generally be used sparingly because it can make programs harder to understand.

---

5. Storage Class Keywords

These control aspects such as **scope, lifetime, and storage behavior** of variables.

```
auto
extern
static
register
```

For example:

```
static int count = 0;
```

---

6. Type Modifier Keywords

These modify the size or range of integer types.

```
short
long
signed
unsigned
```

Example:

```
unsigned int age;
```

An `unsigned` integer cannot store negative values.

7. Structure and User-Defined Type Keywords

`struct`

Used to create a structure.

```
struct Student
{
    int rollno;
    char name[20];
};
```

`union`

Used to define a union.

```
union Data
{
    int i;
    float f;
};
```

`enum`

Used to create named integer constants.

```
enum Day
{
    MONDAY,
    TUESDAY,
    WEDNESDAY
};
```

`typedef`

Creates an alternative name for a type.

```
typedef int number;

number age = 20;
```

8. Other Important Keywords

`const`

Indicates that an object should not be modified through the declared identifier.

```
const int x = 10;
```

`sizeof`

Determines the size in bytes of a type or expression.

```
printf("%zu", sizeof(int));
```

`volatile`

Tells the compiler that a value may change unexpectedly and therefore should not be treated as an ordinary unchanging value for optimization purposes.

```
volatile int value;
```

☆ Very Important: Keyword vs Identifier

This is a common beginner confusion.

Keyword

Reserved by C:

```
int
if
else
while
for
switch
return
```

You **cannot** use these as variable names.

✕ Wrong:

```
int = 10;
```

✕ Wrong:

```
float if;
```

Identifier

A name created by the programmer:

```
age
marks
studentName
total
number
```

Example:

```
int age = 20;
```

Here:

```
int   → Keyword
age   → Identifier
20    → Constant
=     → Assignment operator
;     → Statement terminator
```

Easy Definition for You

You can give them this definition:

"Keywords are reserved words in C that have predefined meanings to the compiler. They cannot be used as names for variables, functions, or other identifiers."

Easy memory concept

Think of keywords as **special words belonging to the C language**.

```
C Language
    ↓
Special Reserved Words
    ↓
Keywords
    ↓
int, if, else, for, while,
switch, case, break, return...
```

Note: If you are using modern C, the exact keyword list depends on the C standard (C90, C99, C11, C17, C23). The **32-keyword list above is the classic list commonly in introductory C courses**.