1. What is Data?

Data means information that we store and use in a computer program.

Real-Life Examples

In a university system, we may need to store:

- Student name
- Age
- Marks
- CGPA
- Grade
- Fee status
- Department
- Registration number

Different types of information require different data types.

---

2. What is a Data Type?

A data type tells C++ what kind of data a variable can store.

Simple Example

Think of a variable as a box.

- Variable = the box
- Data type = label on the box
- Value = information stored inside the box

For example:

```
int age = 20;
```

Here:

int → Data type age → Variable 20 → Value

"int" tells C++ that "age" will store a whole number.

---

3. What is a Variable?

A variable is a named storage location used to store data in a program.

Example

int age = 20;

The variable "age" stores the value "20".

The value of a variable can change:

int age = 20;

age = 21;

Now the value of "age" is "21".

---

4. Why Do We Use Data Types?

C++ needs to know:

1. What type of data a variable will store
2. How the data should be handled
3. How much memory is approximately required

For example:

```
int age = 20;
float cgpa = 3.5;
char grade = 'A';
bool feePaid = true;
```

Each variable stores a different type of information.

---

5. Common C++ Data Types

Data Type| Stores| Example| Real-Life Use "int"| Whole numbers| "25"| Age, marks, students "float"| Decimal numbers| "5.5"| Height, temperature "double"| More precise decimal numbers| "1250.75"| Fees, measurements "char"| One character| "'A'"| Grade, section "string"| Text| ""Ali""| Name, city "bool"| True/False| "true"| Fee paid or not

---

6. `int` — Whole Numbers

Use `int` to store whole numbers without decimal points.

Examples

```
int age = 20;
int students = 500;
int marks = 85;
```

Real-Life Examples

- Age → 20

- Number of students → 500
- Marks → 85
- Number of books → 10

Common Memory

`int` usually uses 4 bytes.

Typical range:

- -2,147,483,648 to 2,147,483,647

---

7. `float` — Decimal Numbers

Use `float` to store numbers with decimal values.

Examples

```
float temperature = 36.5;
float height = 5.4;
float weight = 65.5;
```

Real-Life Examples

- Temperature → 36.5
- Height → 5.4
- Weight → 65.5
- CGPA → 3.5

`float` usually uses 4 bytes.

---

8. `double` — More Precise Decimal Numbers

`double` is used for decimal numbers when we need more precision than `float`.

Example

```
double price = 1250.75;
double pi = 3.1415926535;
```

Real-Life Examples

- Product price → 1250.75
- Distance → 125.6789
- Scientific calculations

`double` usually uses 8 bytes.

> Remember: `float` stores decimal numbers, while `double` stores more precise decimal numbers.

---

9. `char` — One Character

`char` stores one character.

Examples

```
char grade = 'A';
char section = 'B';
```

A `char` value is written inside single quotation marks:

```
'A'
'B'
'X'
```

Do not use double quotation marks for a single `char`:

```
"A"   // This is a string, not a char
```

Real-Life Examples

- Grade → 'A'
- Section → 'B'
- Gender code → 'M'

`char` uses 1 byte.

---

10. `string` — Text

A `string` stores text such as words, names, and sentences.

Examples

```
string name = "Ali";
string university = "Ghazi University";
```

For strings, include:

```
#include <string>
```

Real-Life Examples

```
string studentName = "Ali";
string city = "Dera Ghazi Khan";
string department = "Computer Science";
```

> Remember: `char` stores one character, while `string` stores multiple characters or text.

```
char grade = 'A';
string name = "Ali";
```

---

11. `bool` — True or False

`bool` stores only two logical values:

- `true`
- `false`

Examples

```
bool isStudent = true;
bool feePaid = false;
```

Real-Life Examples

```
bool isAdmitted = true;
bool feePaid = true;
bool attendanceMarked = false;
```

`bool` is useful when a program needs to represent a yes/no or true/false condition.

---

12. Data Types and Memory

Different data types generally require different amounts of memory.

Data Type	Common Memory
`char`	1 byte
`bool`	1 byte*
`short`	2 bytes
`int`	4 bytes
`float`	4 bytes
`double`	8 bytes

Data Type	Common Memory
`long long`	8 bytes

The exact size of some types can depend on the compiler and system.

> Important: Students should first understand why we use different data types. Memory sizes and ranges can be learned afterward.

---

13. Real-Life Example: University Student Record

Imagine we are creating a University Student Record System.

```
string name = "Ali";
int age = 20;
float cgpa = 3.5;
char grade = 'A';
bool feePaid = true;
```

Each data type has a purpose:

- Name → `string` → Text
- Age → `int` → Whole number
- CGPA → `float` → Decimal number
- Grade → `char` → One character
- Fee Paid → `bool` → True/False

This is why we need different data types.

---

14. What Happens If We Use the Wrong Data Type?

Consider:

```
int age = 20;
age = 25.5;
```

`age` is an `int`, so it cannot represent `25.5` exactly.

The decimal part is discarded when the value is converted to `int`.

Therefore, the value becomes:

```
25
```

This demonstrates why choosing the correct data type is important.

---

15. Displaying Variables Using `cout`

`cout` is used to display output on the screen.

Example

```
int age = 20;

cout << age;
```

Output

```
20
```

We can also display text and a variable together:

```
cout << "Age: " << age;
```

Output

```
Age: 20
```

Important Difference

Example 1

```
cout << age;
```

Output:

```
20
```

C++ displays the value stored in the variable.

Example 2

```
cout << "age";
```

Output:

```
age
```

Because `"age"` is text written inside quotation marks.

---

16. Taking Input Using `cin`

`cin` is used to take input from the user.

Example

```
int age;

cout << "Enter your age: ";
cin >> age;

cout << "Your age is: " << age;
```

If the user enters:

```
20
```

The output will be:

```
Your age is: 20
```

---

17. How Variable, Data Type, `cin`, and `cout` Work Together

The basic flow is:

```
Data Type
    ↓
Variable
    ↓
cin → User Input
    ↓
Variable stores the value
    ↓
cout → Display Output
```

For example:

```
int age;
```

```
cout << "Enter your age: ";
cin >> age;

cout << "Your age is: " << age;
```

---

18. Complete Student Information Program

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string name;
    int age;
    float cgpa;

    cout << "Enter your name: ";
    cin >> name;

    cout << "Enter your age: ";
    cin >> age;

    cout << "Enter your CGPA: ";
    cin >> cgpa;

    cout << "\nStudent Information\n";

    cout << "Name: " << name << endl;
    cout << "Age: " << age << endl;
    cout << "CGPA: " << cgpa << endl;

    return 0;
}
```

What does this program teach?

Data Type ↓ Variable ↓ User Input using cin ↓ Value stored in variable ↓ Output using cout

---

19. Predict Before You Run

A useful programming learning technique is:

«Predict the output before running the program.»

Example 1

int marks = 85;

cout << marks;

Ask yourself:

What will be displayed?

Answer:

85

Example 2

float cgpa = 3.75;

cout << cgpa;

Output:

3.75

Example 3

char grade = 'A';

cout << grade;

Output:

A

---

20. Quick Revision

Data Type

«A data type tells C++ what kind of data a variable can store.»

Variable

«A variable is a named storage location used to store data.»

"int"

«Stores whole numbers.»

"float"

«Stores decimal numbers.»

"double"

«Stores decimal numbers with greater precision.»

"char"

«Stores one character.»

"string"

«Stores text.»

"bool"

«Stores "true" or "false".»

"cin"

«Takes input from the user.»

"cout"

«Displays output on the screen.»

---

21. Easy Way to Remember

Think about a University Student Admission System:

Student Name → string Age → int Fee Amount → double CGPA → float Grade → char Admission Completed → bool

Complete Concept

DATA ↓ DATA TYPE ↓ VARIABLE ↓ VALUE ↓ cin → Input ↓ cout → Output

Recommended Learning Sequence

Learn these concepts in this order:

Data → Variable → Data Type → Value → Memory → "cout" → "cin" → Practical Program → Practice