Variables And Data Types in C++:

Variables: Named Storage Memory location

All variables use data type during declaration to restrict the type of data to be stored. Therefore, we can say that data types are used to tell the variables the type of data they can store. Whenever a variable is defined in C++, the compiler allocates some memory for that variable based on the data type with which it is declared. Every data type requires a different amount of memory.

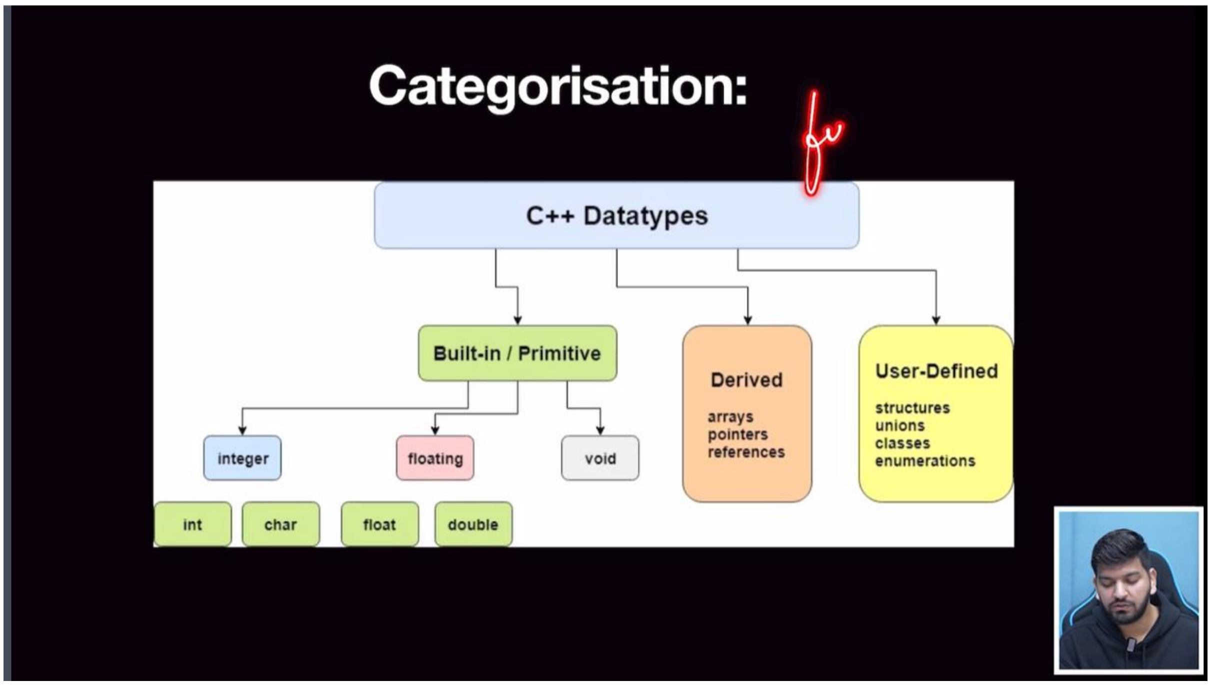
C++ supports a wide variety of data types and the programmer can select the data type appropriate to the needs of the application. Data types specify the size and types of values to be stored. However, storage representation and machine instructions to manipulate each data type differ from machine to machine, although C++ instructions are identical on all machines.

C++ supports the following data types:

Primary or Built-in or Fundamental data type

Derived data types

User-defined data types



Program:

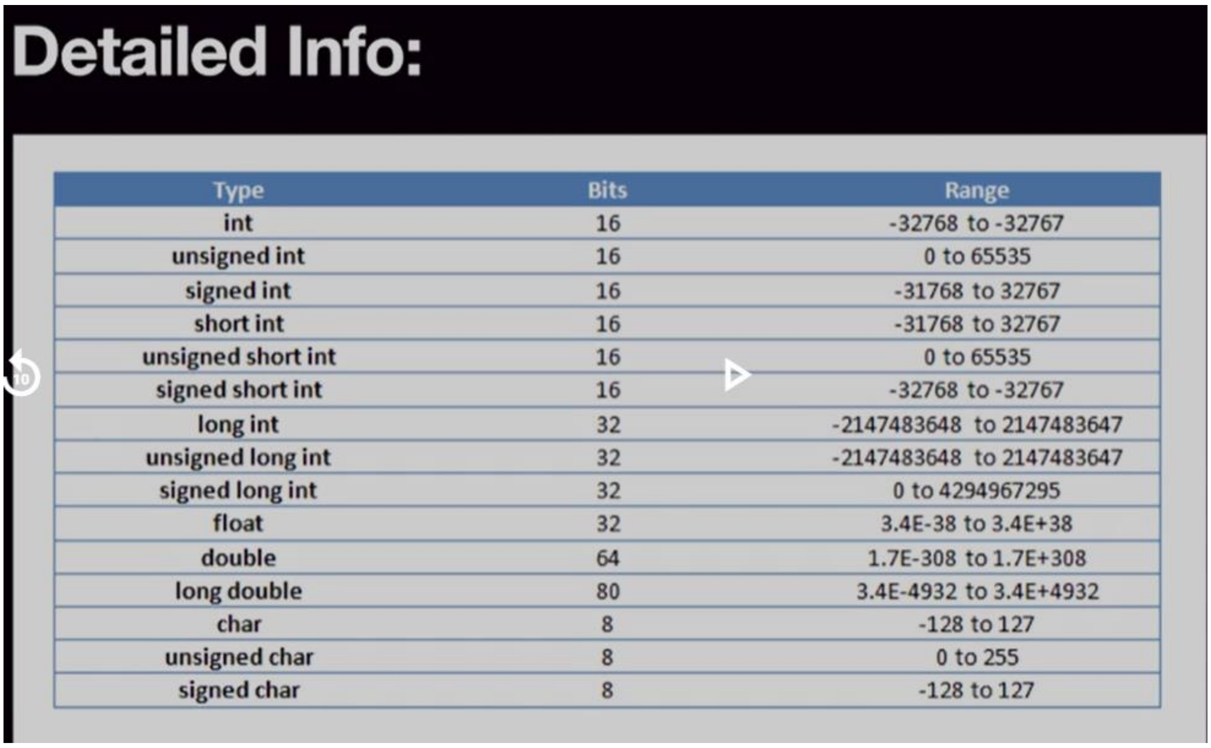
```
#include<iostream>

using namespace std;


int main(){
    //Variables and Data Types
```

```
int a; //Variable decelaration

a=10; //Value assignment


int b = 20; //Variable deceleration and assignment


int count = 50;


cout<<"count(Integer): "<<count<<endl;

cout<<"size of Integer Data type :"<<sizeof(int)<<endl<<endl;


float share = 3.14;


cout<<"share(float): "<<share<<endl;

cout<<"size of Float Data type :"<<sizeof(float)<<endl<<endl;


char  alphabet = 'A';


cout<<"alphabet(char): "<<alphabet<<endl;

cout<<"size of char Data type :"<<sizeof(char)<<endl<<endl;


double weight = 55.0947554;


cout<<"weight(Double): "<<weight<<endl;

cout<<"size of Double Data type :"<<sizeof(double)<<endl<<endl;


bool isFemale = true;


cout<<"isFemale(bool): "<<isFemale<<endl;

cout<<"size of bool Data type :"<<sizeof(bool)<<endl<<endl;


bool isMale = false;
```

```
    bool isTrue = 1;
    bool isFalse = 0;

    cout<<isFemale<<" "<<isMale<<" "<<isTrue<<" "<<isFalse<<endl;

}
```

Output :

```
PS E:\C++ PROGRAMMES> g++ datatypes.cpp
PS E:\C++ PROGRAMMES> ./a.exe
count(Integer): 50
size of Integer Data type :4

share(float): 3.14
size of Float Data type :4

alphabet(char): A
size of char Data type :1

weight(Double): 55.0948
size of Double Data type :8

isFemale(bool): 1
size of bool Data type :1

1 0 1 0
```



Detailed Info:

Type	Bits	Range
int	16	-32768 to -32767
unsigned int	16	0 to 65535
signed int	16	-31768 to 32767
short int	16	-31768 to 32767
unsigned short int	16	0 to 65535
signed short int	16	-32768 to -32767
long int	32	-2147483648 to 2147483647
unsigned long int	32	-2147483648 to 2147483647
signed long int	32	0 to 4294967295
float	32	3.4E-38 to 3.4E+38
double	64	1.7E-308 to 1.7E+308
long double	80	3.4E-4932 to 3.4E+4932
char	8	-128 to 127
unsigned char	8	0 to 255
signed char	8	-128 to 127

Sizeof operator in C++:

The sizeof operator is a unary compile-time operator used to determine the size of variables, data types, and constants in bytes at compile time. It can also determine the size of classes, structures, and unions.

sizeof (data type)   or sizeof (expression)

Home Work:

1.why double data in c++ is automatically round off?

Computers store floating-point numbers (like double) in binary format, using a finite number of bits (typically 64 for double). Decimal numbers don't always have exact representations in binary, especially those with infinite decimal expansions (e.g., 0.1, 1/3).To fit these numbers into limited memory, some precision must be sacrificed, leading to rounding.

2. How the data types stored inside memory

All datatypes are stored the data in binary format

Integers (int):

Stored directly as binary values at specific memory addresses.

Size varies depending on system architecture (e.g., 32 bits or 64 bits).

Characters (char):

Represented as single bytes using a character encoding (e.g., ASCII).

Each byte stores the numerical code for a character.

Floating-point numbers (float, double):

Use the IEEE 754 standard for binary representation.

Divided into sign bit, exponent, and mantissa for precision and range.

Double-precision (double) offers more precision than single-precision (float).

3.How signed and unsigned data is stored in memory

Storage Method:

Binary Representation: Both use binary format (base 2).

Bit Allocation:

- Unsigned: All bits represent the magnitude (value).
- Signed: One bit (usually the most significant bit, MSB) indicates sign:
  - 0 = positive
  - 1 = negative

Two's Complement (Common for Signed Integers):

- Negative numbers are represented by inverting all bits and adding 1.

Example:

4-bit unsigned: 0100 represents 4

4-bit signed: 0100 represents 4, but 1100 represents -4

4.Conversion from Decimal to Binary and Binary to decimal

Decimal to Binary:

- Repeated Division by 2:
  - Divide the decimal number by 2 repeatedly, recording the remainders (0 or 1) in reverse order.
  - Continue until the quotient becomes 0.
  - The binary representation is the sequence of remainders.
  - Example: 13 in decimal to binary:
  - 13 / 2 = 6 remainder 1
  - 6 / 2 = 3 remainder 0

3 / 2 = 1 remainder 1

1 / 2 = 0 remainder 1

Binary: 1101

Binary to Decimal:

Positional Notation:

Each binary digit (0 or 1) has a place value based on its position, starting from 2^0 on the right, increasing to 2^1, 2^2, and so on.

Multiply each digit by its corresponding power of 2.

Add up the products to get the decimal value.

Example: 1101 in binary to decimal:

(1 x 2^3) + (1 x 2^2) + (0 x 2^1) + (1 x 2^0)

= 8 + 4 + 0 + 1

= 13

Additional Points:

Integer vs. Fractional Parts: For decimal numbers with fractional parts, the conversion involves separate processes for the integer and fractional parts.

For the fractional part, multiply the decimal fraction by 2 repeatedly, recording the integer part of each product as a binary digit, until the fractional part becomes 0 or the desired precision is reached.

Floating-Point Numbers: Conversion for floating-point numbers (like float and double) is more complex due to their IEEE 754 format, involving separate representations for sign, exponent, and mantissa.