Week 1

Session 1: Basic data types - characters, integers and floating numbers

Element 1: Recognize simple data types, reserved words, and declaration and executable statements

ECT 124: Writing Programs using C++

Performance criteria (PC) for E1

PC1: Identify simple data types such as characters, integers, and floating numbers.

In this lesson!

PC2: Specify the functionality of reserved words in C++.

PC3: Differentiate between declarative and executable statements in C++.

Learning objectives:

By the end of this lesson, the student should be able to:

- ✓ Understand the concept of data types in C++.
- ✓ Identify various simple data types such as char, int, and float.
- ✓ Demonstrate the usages of these data types for C++ programming.

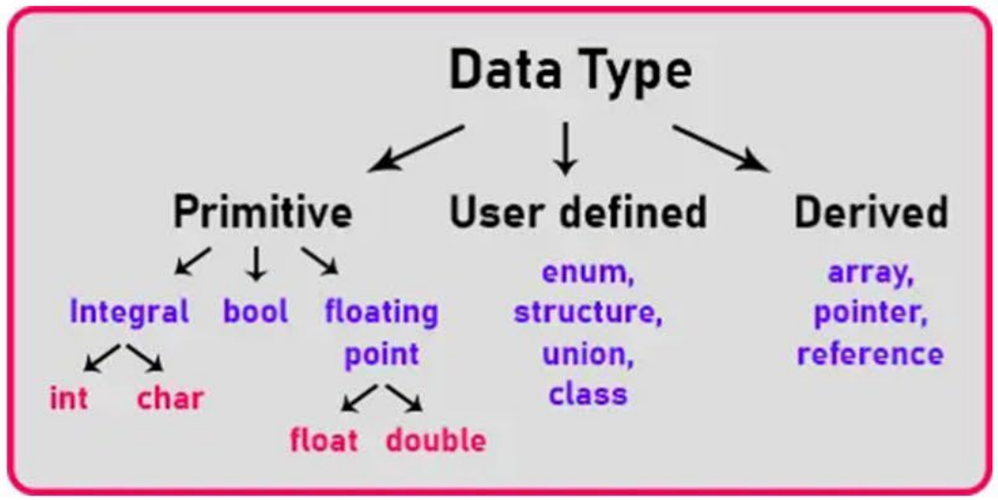
Class Activity 1

Lesson 4

Video

- C++ program provides a numerous set of data types depending on the nature, size, and associated operations. Basically, the data types are classified into fundamental or primitive data types, derived data types, and user-defined data types.

Set of Data Types	Description	Examples
User-defined Data Types	Flexibility for programmers to define their own data types	class, union, struct, enum,
Fundamental/Primitive Data Types	Pre-defined or built-in data types defined in the compiler with certain size or memory space allocation.	char, int, float
Derived Data Types	Convenient way for programmers to group all the information for a particular item. Constructed from fundamental data types through grouping or alteration in the size.	arrays, pointers, functions



Fundamental Data Types

- Fundamental data types also known as the primary data types since they are pre-defined or already exist in the C++ language.

- All the other types of data types (derived and user-defined data types) are derived from the fundamental data types.

- The fundamental data types in C++ program are of 4 types: **int**, **char**, **float**, and **double**. The table represents the memory consumed or size of each fundamental data types.

Fundamental Data Type	Purpose	Memory	Range
char	Stores a single letter, number or symbols.	1 byte	-128 to 127
int	Stores whole numbers.	4 bytes	± 2,147,483,648
float	Stores fractional numbers, containing one or more decimals. Typically 6-7 decimal digits.	4 bytes	1.2E-38 to 3.4E+38
double	Stores fractional numbers, containing one or more decimals. Typically 15 decimal digits.	8 bytes	2.3E-308 to 1.7E+308

char

- Characters are the symbols covered by the character set of the C++ language. All letters, digits, special symbols, punctuations, etc. come under this category. When these characters are used as data they are considered as char data type in C++ program.

- Typically, computer works using the binary number system that is 0 and 1. Thus, a character should also be represented in 0 and 1 using a certain character codes.

- The character codes are called ASCII (American standard code for information interchange) codes. Every alphabet or letter and special symbols in the English language are represented by a certain given code in the following table.

- The C++ program syntax for char data type: **char variable_name;**

- The char data type is used to store the characters. The characters stored in a variable of char data type have a specific value equivalent to a corresponding integer code.

ASCII codes

Space	32	0	48	@	64	P	80	`	96	p	112
!	33	1	49	A	65	Q	81	a	97	q	113
"	34	2	50	B	66	R	82	b	98	r	114
#	35	3	51	C	67	S	83	c	99	s	115
$	36	4	52	D	68	T	84	d	100	t	116
%	37	5	53	E	69	U	85	e	101	u	117
&	38	6	54	F	70	V	86	f	102	v	118
'	39	7	55	G	71	W	87	g	103	w	119
(	40	8	56	H	72	X	88	h	104	x	120
)	41	9	57	I	73	Y	89	i	105	y	121
*	42	:	58	J	74	Z	90	j	106	z	122
+	43	;	59	K	75	[	91	k	107	{	123
,	44	>	60	L	76	\	92	l	108	|	124
-	45	=	61	M	77	]	93	m	109	}	125
.	46	<	62	N	78	^	94	n	110	~	126
/	47	?	63	O	79	_	95	o	111	del	127

- For example, capital A is represented as number 65, then B is represented as number 66. So, an integer number is used for representing the character. Only on the screen, the printing will be done as A but inside the memory, it is number 65.

- The ASCII code for upper case characters A to Z starts from integer numbers 65 to 90 and for lower case characters a to z, it starts from integer numbers 97 to 122.

- It is in binary form (converted from the integer number to binary number) that all the codes are used by the computers for representing characters.

Escape Sequences

- Nongraphic characters can be expressed by means of escape sequences, for example \t, which represents a tab.
- A \n will insert an empty line before printing the following statement using a new line.
- An escape sequence always begins with a \ (backslash) and represents a single character.

Single character	Meaning
\a	alert (BEL)
\b	backspace (BS)
\t	horizontal tab (HT)
\n	line feed (LF)
\v	vertical tab (VT)
\f	form feed (FF)
\r	carriage return (CR)
\"	" (double quote)
\'	' (single quote)
\?	? (question mark)
\\	\ (backslash)

Example program **WITHOUT** escape sequences

```
1  /*Sample program without escape sequence*/
2  #include <iostream>
3  using namespace std;
4  int main()
5  {
6  cout << "This is a string"
7  " with many escape sequences!";
8  return 0;
9  }
```

```
C:\Users\msharizal\Documents\HCT_Aug 2019\01_Lecture Materials and Details\11
This is a string with many escape sequences!
--------------------------------
Process exited after 0.04162 seconds with return value 0
Press any key to continue . . .
```

Example program **WITH** escape sequences

```
1  /*Sample program without escape sequence*/
2  #include <iostream>
3  using namespace std;
4  int main()
5  {
6  cout << "\nThis is\t a string\n\t\t"
7  " with \"many\" escape sequences!\n";
8  return 0;
9  }
```

```
C:\Users\msharizal\Documents\HCT_Aug 2019\01_Lecture Materials and Details\

This is  a string
                with "many" escape sequences!

--------------------------------
Process exited after 0.03969 seconds with return value 0
Press any key to continue . . .
```

int

- Integers are whole numbers without a fractional part. They can be positive, zero or negative. The keyword **int** represents integer numbers within a specific range.

- The int data type is used to store the integer values. In C++ language, the integers cannot be represented as floating or decimal point numbers. If the operation of two integers results in a fraction number, the integer part will be stored as the final result.

- The C++ program syntax for int data type: **int variable_name;**

- The types **short**, **int**, and **long** are available for operations with integers. These types are distinguished by their ranges of values.

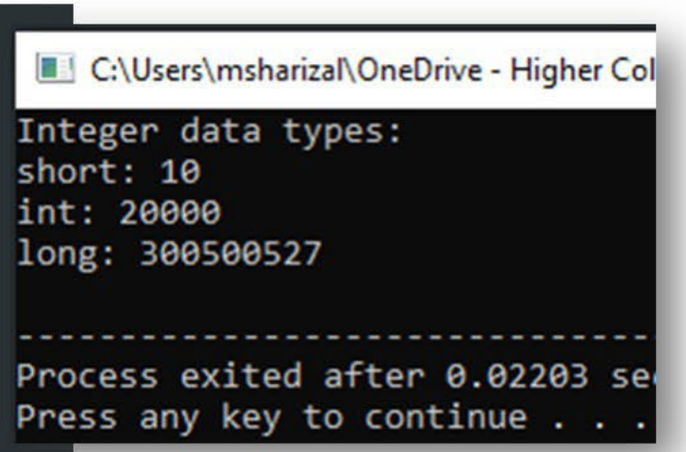
Example 2

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

int main()
{
  // Integer data types
  short a = 10;
  int b = 20000;
  long c = 300500527;

  cout << "Integer data types: " << endl;
  cout << "short: " << a << endl;
  cout << "int: " << b << endl;
  cout << "long: " << c << endl;

  return 0;
}
```



float

- In C++ program, a number with a fraction part is called a floating-point type. The float data type is used to store the floating-point numbers, for example, 5.0, 7.27, -30.3, and -2.55.

- It should be noticed carefully that 5 is an integer but 5.0 is a floating-point number. Although the value of both the digits is the same, 5.0 has a decimal number which differentiates it with 5.

- There are three different floating-point sizes: **float**, **double**, and **long double**. An example of a C++ syntax for float is: **float variable_name;**

- The **float** and **double** are used to store floating-point numbers (decimals and exponentials). Usually, the size of **float** is 4 bytes and the size of **double** is 8 bytes. Hence, double has two times the precision of float.

- As with integer types, the differences for example between **float** and **double** involve the range of their values and their storage requirements. If you cannot decide between the types **float** and **double**, use **double**.

Example 3

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

int main()
{
  // Floating-point data types
  float e = 3.1;
  double f = 3.14;
  long double g = 3.14159;
  float h = 2.5e-12;
  cout << "Floating-point data types: " << endl;
  cout << "float: " << e << endl;
  cout << "double: " << f << endl;
  cout << "long double: " << g << endl;
  cout << "float exponential: " << h << endl;

  return 0;
}
```

```
C:\Users\msharizal\OneDrive - Higher College
Floating-point data types:
float: 3.1
double: 3.14
long double: 3.14159
float exponential: 2.5e-012

--------------------------------
Process exited after 0.04303 secon
Press any key to continue . . .
```

Class Activity 2

Write the C++ program below using the online compiler

(https://www.onlinegdb.com/online_c++_compiler) and paste the screenshot of the

program output at the following slide.

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

int main()
{
    int a = 5;

    float b = 7.5;

    double c = 5.34752;

    char d = 'C';

    cout <<"Integer value is = "<< a <<endl;
    cout <<"Float value is = "<< b << endl;
    cout <<"Double value is = "<< c <<endl;
    cout <<"Char value is = "<< d <<endl;

    return 0;
}
```

Class Activity 2

Paste the screenshot of your program output.

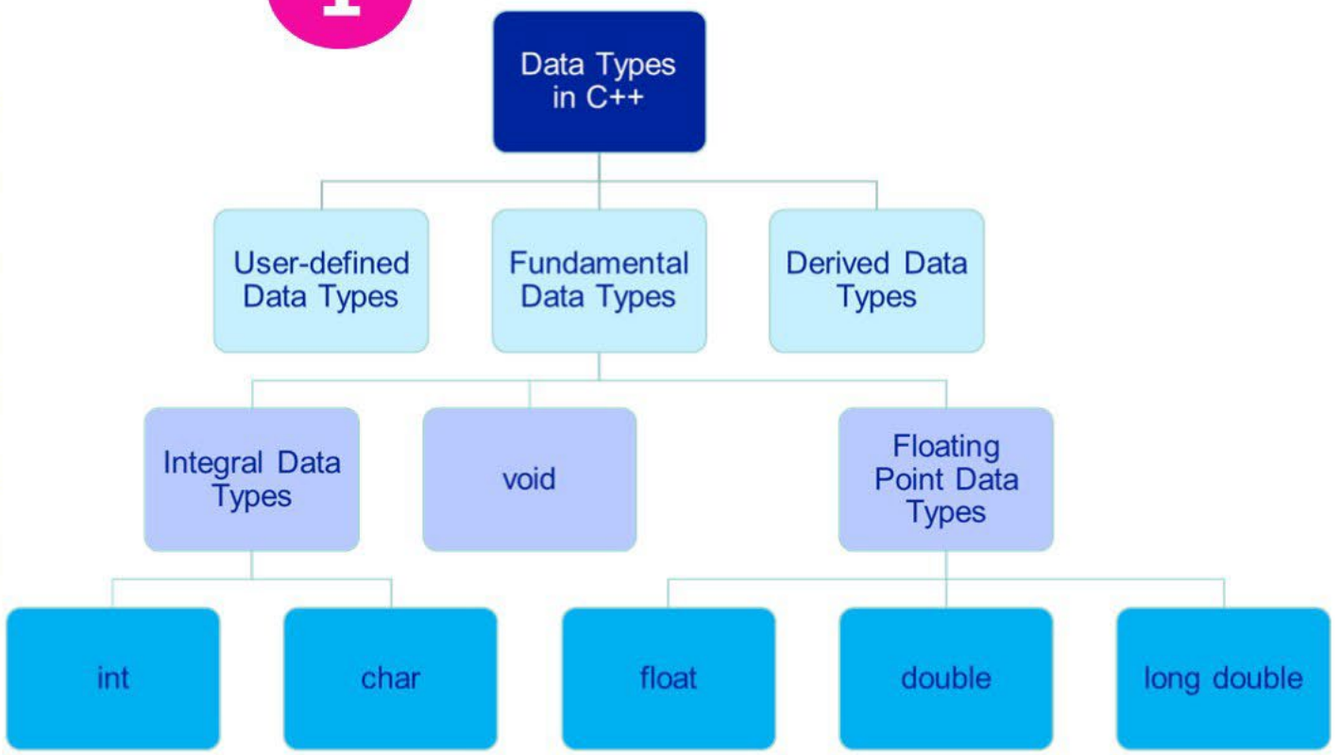
Instructions

Collaborate Board

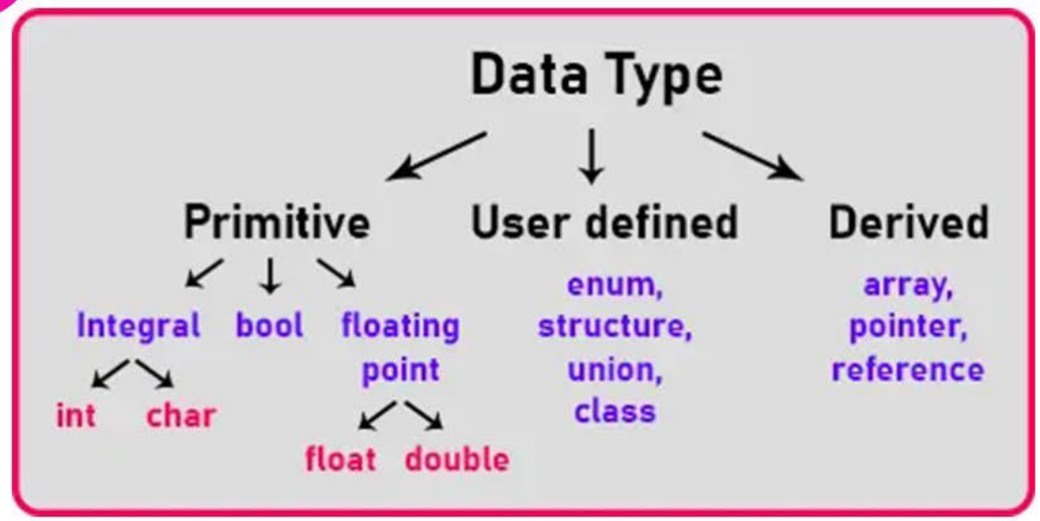
Class Activity 2

Summary

1


Data Types in C++
User-defined Data Types
Fundamental Data Types
Derived Data Types
Integral Data Types
void
Floating Point Data Types
int
char
float
double
long double

2


Data Type
Primitive
User defined
Derived
Integral
bool
floating point
enum, structure, union, class
array, pointer, reference
int
char
float
double

Higher Colleges of Technology

كليات التقنية العليا

Thank You

800 MyHCT (800 69428) www.hct.ac.ae HCT_UAE | hctuae