Computer Programming (a)

E1123

Fall 2022-2023

Lecture 3

C++ Fundamental Operations

INSTRUCTOR

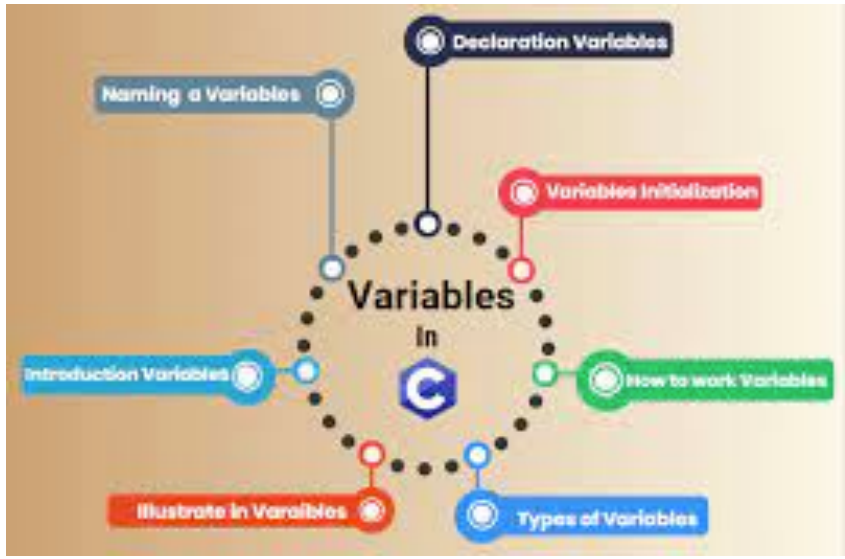
DR / AYMAN SOLIMAN

➢ Contents

1) Variables and Assignments
2) Identifiers
3) C++ keywords
4) Whitespace and basic formatting
5) Data types
6) Arithmetic Operators and Operator Precedence
7) Allocating Memory with Constants and Variables
8) Assignment Statement
9) Declaring & Initializing Variables

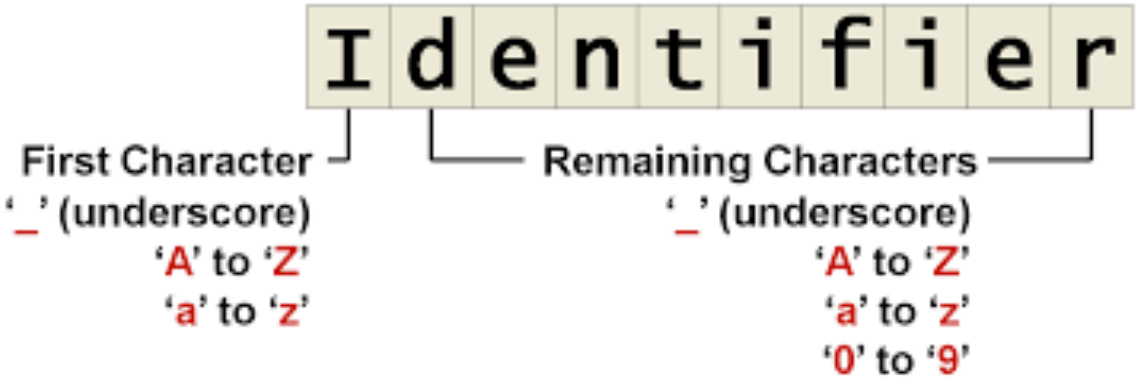
❑ Variables and Assignments

- Variables are like small blackboards
  - ❑ We can write a number on them
  - ❑ We can change the number
  - ❑ We can erase the number
- C++ variables are names for memory locations
  - ❑ We can write a value in them
  - ❑ We can change the value stored there
  - ❑ We cannot erase the memory location



❑ identifiers

- Variables names are called identifiers
- Choosing variable names
  - ❑ Use short meaningful names that represent data to be stored
  - ❑ generally, avoid single letter variables
- First character must be
  - ❑ a letter
  - ❑ the underscore character
- Remaining characters must be
  - ❑ letters
  - ❑ numbers
  - ❑ underscore character



- Identifiers can not be any keywords (reserved words)

❑ C++ keywords

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

- Keywords are words reserved as part of the language
- They cannot be used by the programmer to name things
- They consist of lowercase letters only
- They have special meaning to the compiler

❑ Whitespace and basic formatting

- **Whitespace** is a term that refers to characters that are used for formatting purposes. In C++, this refers primarily to spaces, tabs, and (sometimes) newlines.

- The C++ compiler generally ignores whitespace, with a few minor exceptions. **The following statements all do the exact same thing:**

```
cout << "Hello world!";
```

```
cout          <<          "Hello world!";
```

```
                          cout << "Hello world!“     ;
```

```
cout
<< "Hello world!";
```

```
cout << "Hello ";  cout << "world!";
```

```
int main() { return 0; }
```

```
int main() {
   return 0; }
```

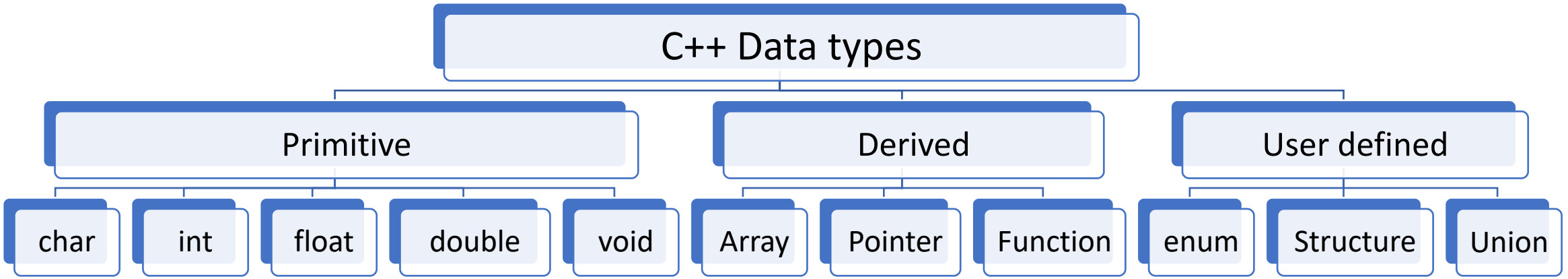
```
int main()
{   return 0; }
```

```
int main()
{
   return 0;
}
```

❑ Data types

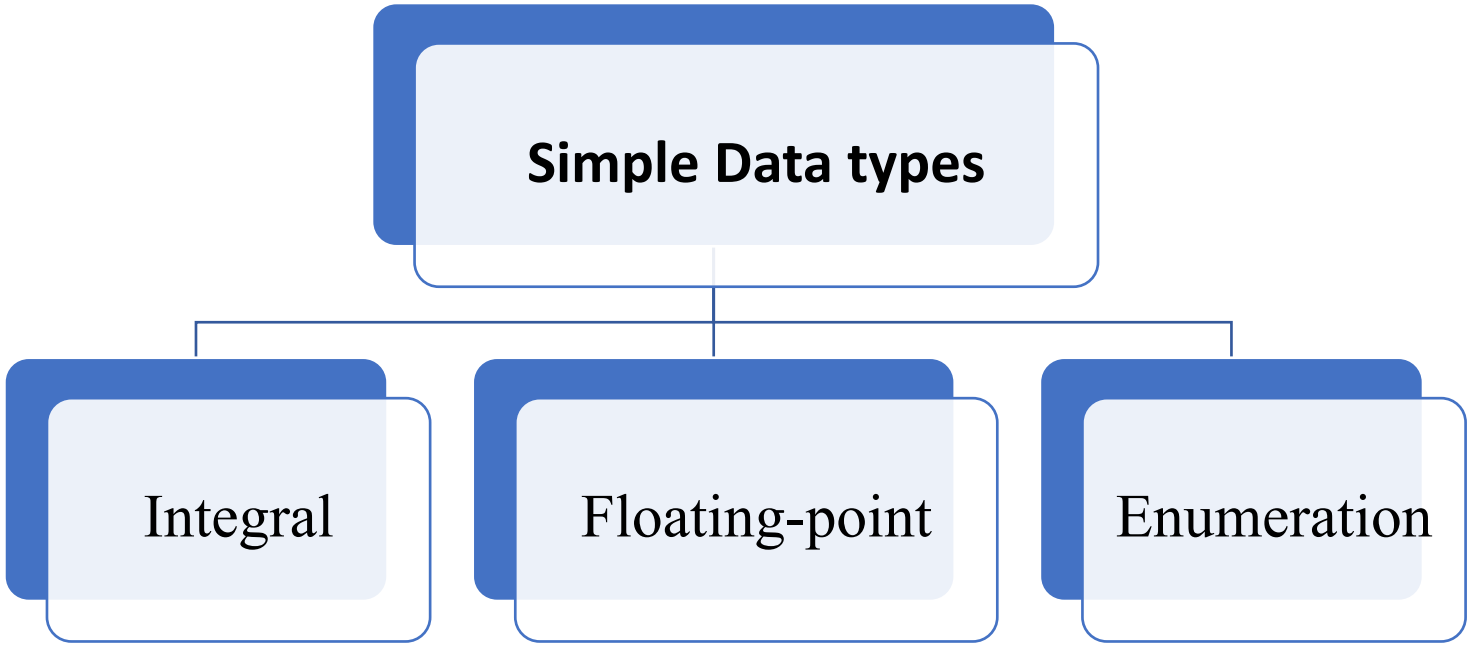
- Data type: set of values together with a set of operations

- C++ data types fall into three categories:



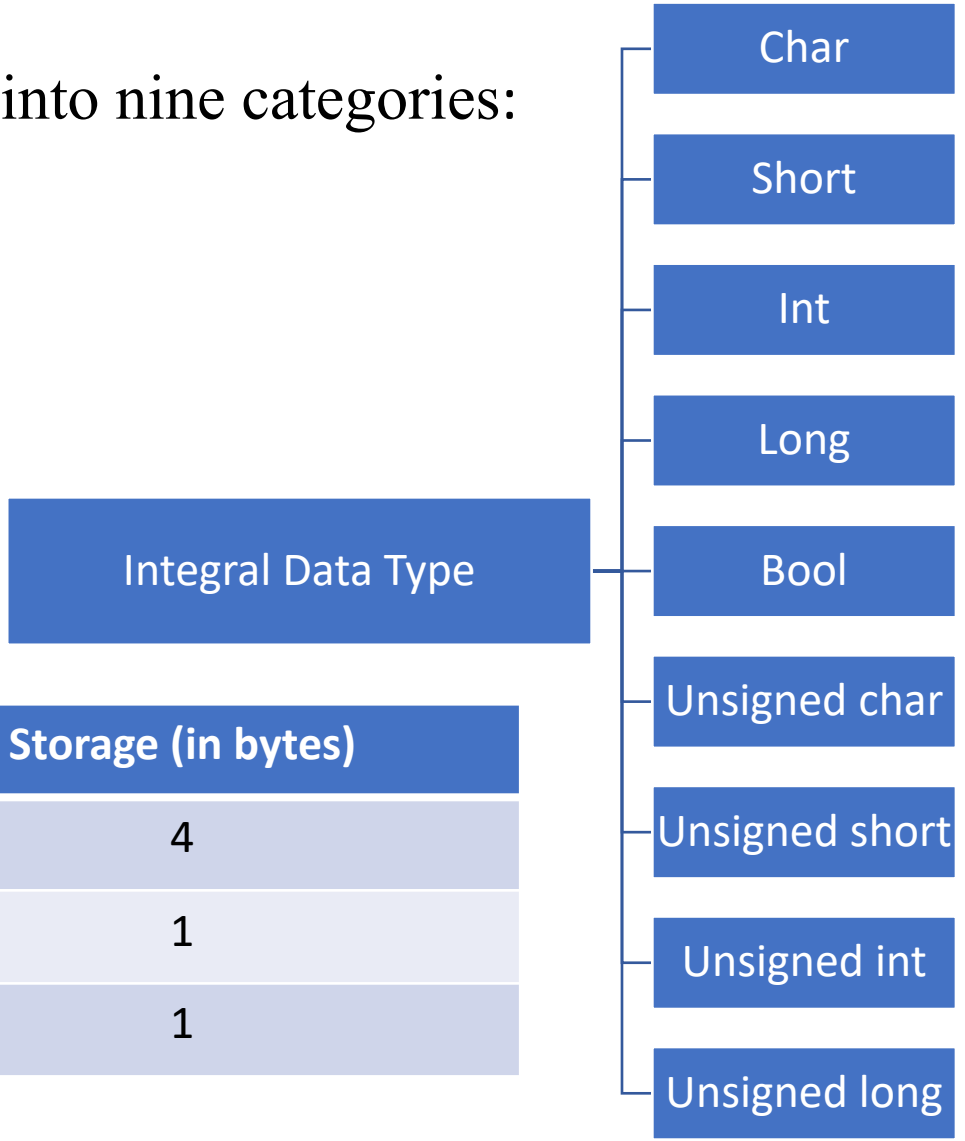
❑ Simple Data types

- Three categories of simple data
  - ❑ Integral: integers (numbers without a decimal)
  - ❑ Floating-point: decimal numbers
  - ❑ Enumeration type: user-defined data type



❑ Simple Data types (cont.)

- Integral data types are further classified into nine categories:



Data Types	Values	Storage (in bytes)
int	-2147483648 to 2147483647	4
bool	true or false	1
char	-128 to 127	1

❑ int Data Type

- Examples:

  -6728

  0

  78

  +763

- Positive integers do not need a + sign

- No commas are used within an integer

- Commas are used for separating items in a list

❑ bool Data Type

bool type

Two values: true and false

Manipulate logical (Boolean) expressions

- true and false are called logical values
- bool, true, and false are reserved words

❑ Char Data Type

The smallest integral data type

Used for characters: letters, digits, and special symbols

Each character is enclosed in single quotes

'A', 'a', '0', '*', '+', '$', '&'

A blank space is a character and is written ' ', with a space left between the single quotes

❑ floating-point Data Type

- C++ uses scientific notation to represent real numbers (floating-point notation)

Real Number	C++ Floating-Point Notation
75.924	7.592400E1
0.18	1.800000E-1
0.0000453	4.530000E-5
-1.482	-1.482000E0
7800.0	7.800000E3

❑ floating-point Data Type (cont.)

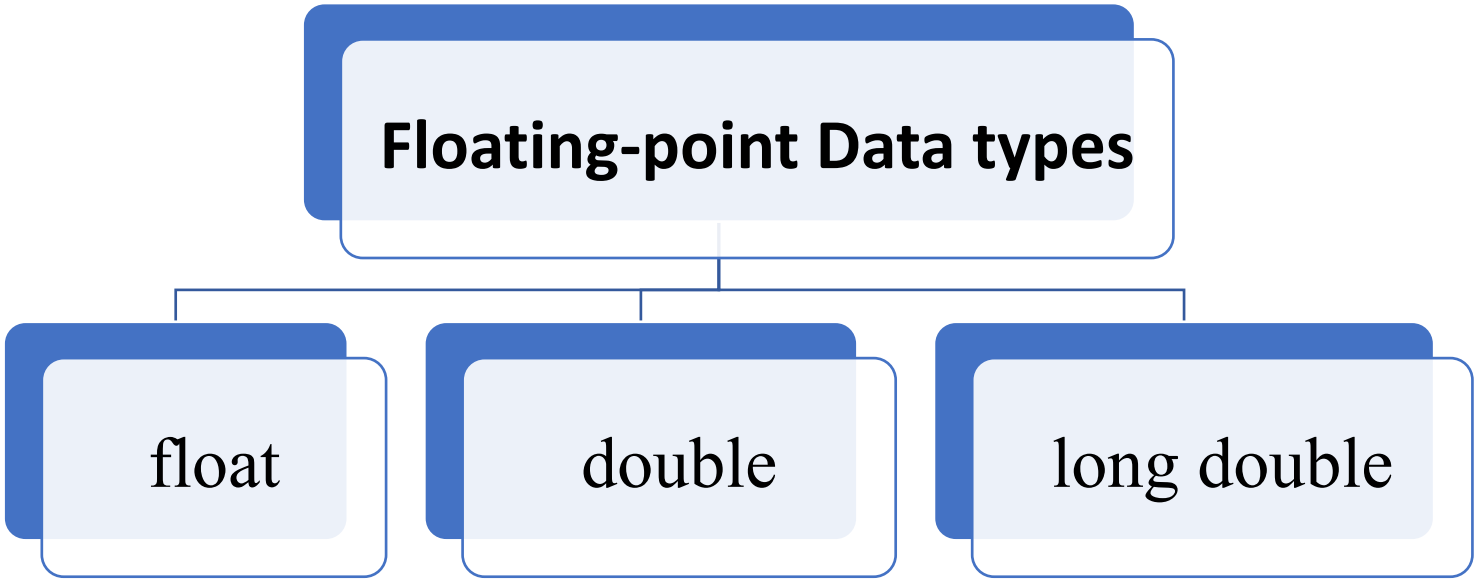
float: represents any real number

Range: -3.4E+38 to 3.4E+38 (Four Bytes)

double: represents any real number

Range: -1.7E+308 to 1.7E+308 (Eight Bytes)

On most newer compilers, data types double and long double are same

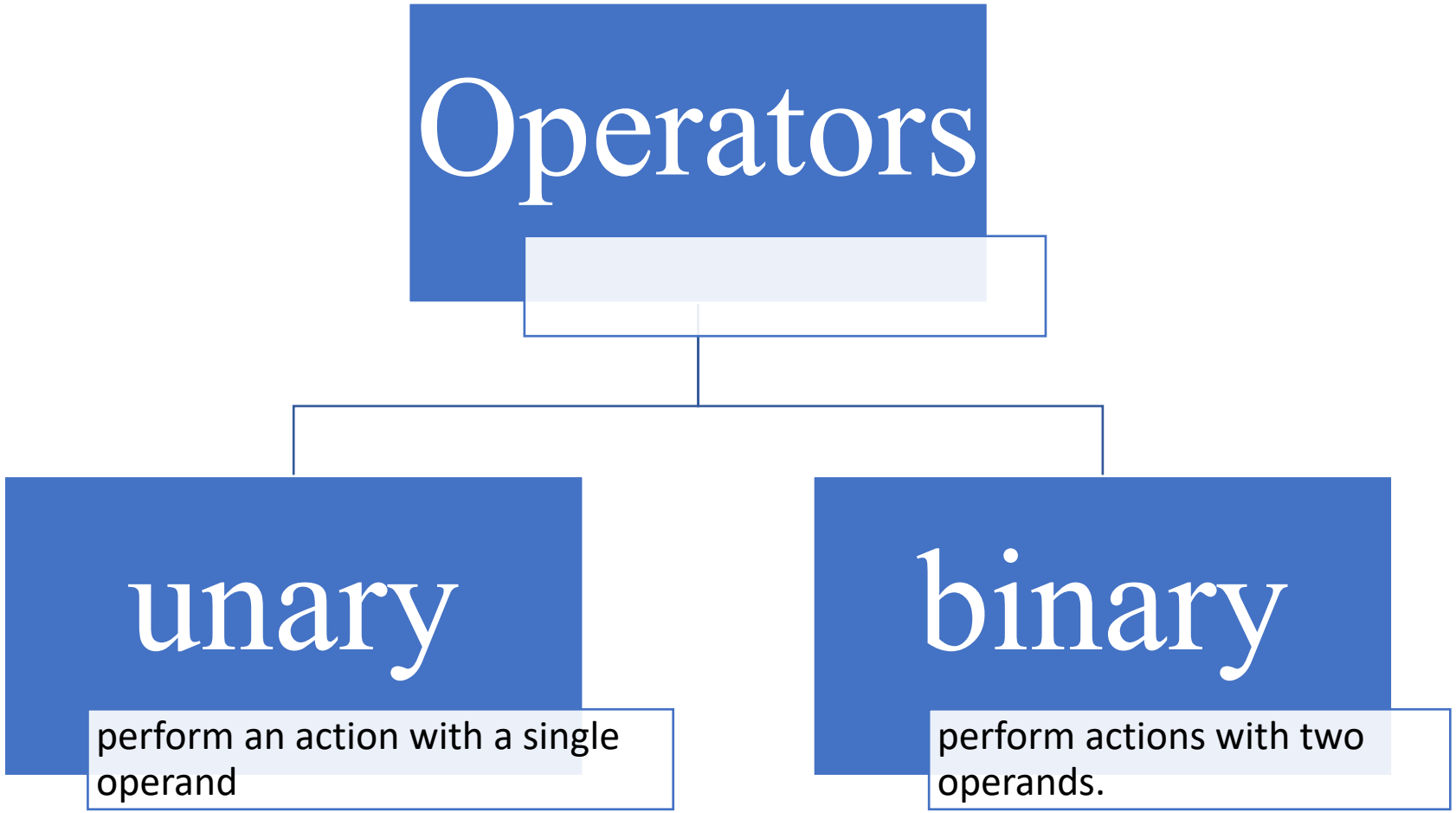


❑ Arithmetic Operators and Operator Precedence

- C++ arithmetic operators:
  - ❑ + addition
  - ❑ - subtraction
  - ❑ * multiplication
  - ❑ / division
  - ❑ % modulus operator

- +, -, *, and / can be used with integral and floating-point data types

- Operators can be unary or binary


Operators
unary
perform an action with a single operand
binary
perform actions with two operands.

❑ Order of Precedence

- All operations inside of () are evaluated first
- *, /, and % are at the same level of precedence and are evaluated next
- + and – have the same level of precedence and are evaluated last
- When operators are on the same level
  - Performed from left to right (associativity)
- 3 * 7 - 6 + 2 * 5 / 4 + 6 means
  - (((3 * 7) – 6) + ((2 * 5) / 4)) + 6

❑ Allocating Memory with Constants and Variables

Named constant: memory location whose content can't change during execution

The syntax to declare a named constant is:

In C++, const is a reserved word

Consider the following C++ statements:

```
const double CONVERSION = 2.54;
const int NO_OF_STUDENTS = 20;
const char BLANK = ' ';
const double PAY_RATE = 15.75;
```

```
const dataType identifier = value;
```

Variable: memory location whose content may change during execution

The syntax to declare a named constant is:

```
double amountDue;
int counter;
char ch;
int x, y;
string name;
```

```
dataType identifier, identifier, . . .;
```

❑ Assignment Statement

The assignment statement takes the form:

Variable = expression;

Expression is evaluated and its value is assigned to the variable on the left side

In C++, = is called the assignment operator

```
int num1, num2;
double sale;
char first;
string str;
num1 = 4;
num2 = 4 * 5 - 11;
sale = 0.02 * 1000;
first = 'D';
str = "It is a sunny day.";
```

```
1. num1 = 18;
2. num1 = num1 + 27;
3. num2 = num1;
4. num3 = num2 / 5;
5. num3 = num3 / 4;
```

❑ Declaring & Initializing Variables

- Variables can be initialized when declared:

```
int first=13, second=10;
char ch=' ';
double x=12.6;
```

- All variables must be initialized before they are used

  But not necessarily during declaration

Thank you