Tishk International University
Faculty of Applied Science
Information Technology Department

Variables

Programming I

Lecture 3

Fall 2025

Course Code: IT117

Grade 1

Islam Abdulazeez
islam.abdulaziz@tiu.edu.iq

December 15, 2025

Outlines

- ✓ Introduction to variables
- ✓ C++ basic data types
- ✓ Variable declaration and initialization
- ✓ Data type selection and ranges
- ✓ Overflow, underflow, and naming rules
- ✓ Constants and variable scope

Learning Outcomes

- **At the end of today's session, you will be able to:**

  - ✓ Define variables and common C++ data types.
  - ✓ Explain differences between basic data types.
  - ✓ Declare and initialize variables correctly.
  - ✓ Select appropriate data types for simple programming tasks.

C++ Variables

- Variables are containers for storing data values.
- In C++, there are different types of variables (defined with different keywords), for example:

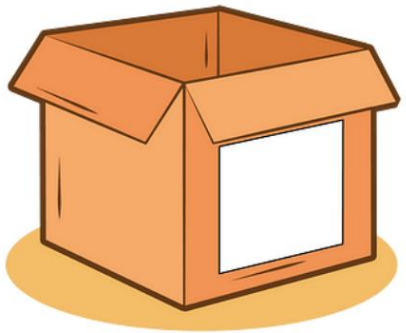
✓ **int** - Stores integers (whole numbers), without decimals, such as 123 or -123

✓ **double**, **float** - Stores floating point numbers, with decimals, such as 19.99 or -19.99

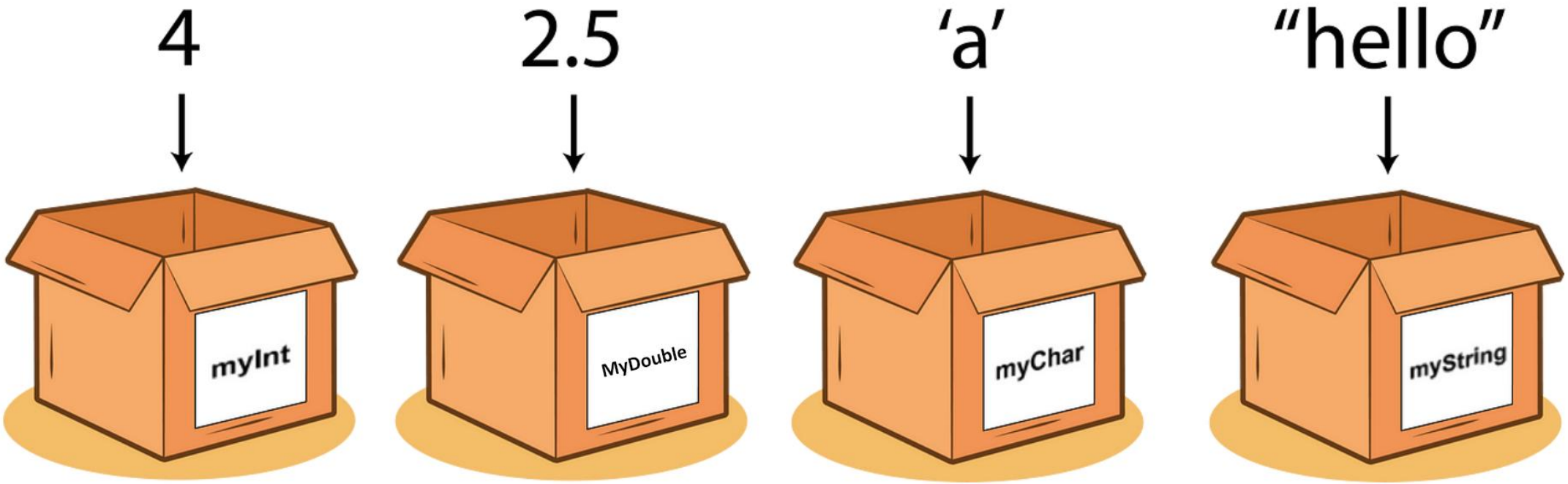
✓ **char** - Stores single characters, such as 'a' or 'B’ . Char values are surrounded by single quotes

✓ **string** - Stores text, such as "Hello World". String values are surrounded by double quotes

✓ **bool** - Stores values with two states: true or false



C++ Variables



Sample Data Type Matches

- **int** - Number of students in class
- **double** - Your GPA (3.85)
- **char** - Your grade in Programming I ('A')
- **string** - Your email address
- **bool** - Is it raining? (true/false)

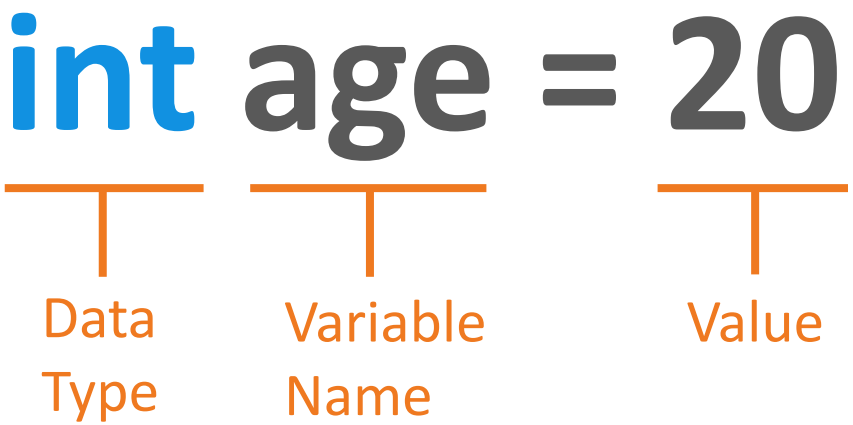
Declaration with Initialization

- data_type variable_name; // Declaration

  variable_name = **value**; // Initialization

- data_type variable_name = **value**; // Declaration and initialization

int Data Type



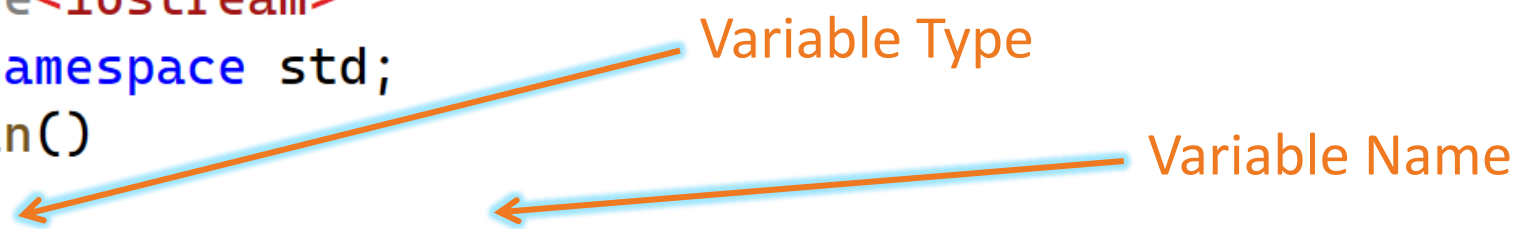
int Data Type

- The int data type is used for storing whole numbers without decimals. It's suitable for representing quantities that are counted in whole units.

```
#include<iostream>
using namespace std;
int main()
{
    int numberOfStudents = 52;

    int x;
    x = 5;

    return 0;
}
```



int Data Type

```
#include<iostream>
using namespace std;
int main()
{
    int numberOfStudents = 52;

    int x;
    x = 5;

    return 0;
}
```

Declaration and Initialization together

Declaring a variable

Initializing a variable

Multiple Variables

```
int score1 = 95, score2 = 87, score3;
```

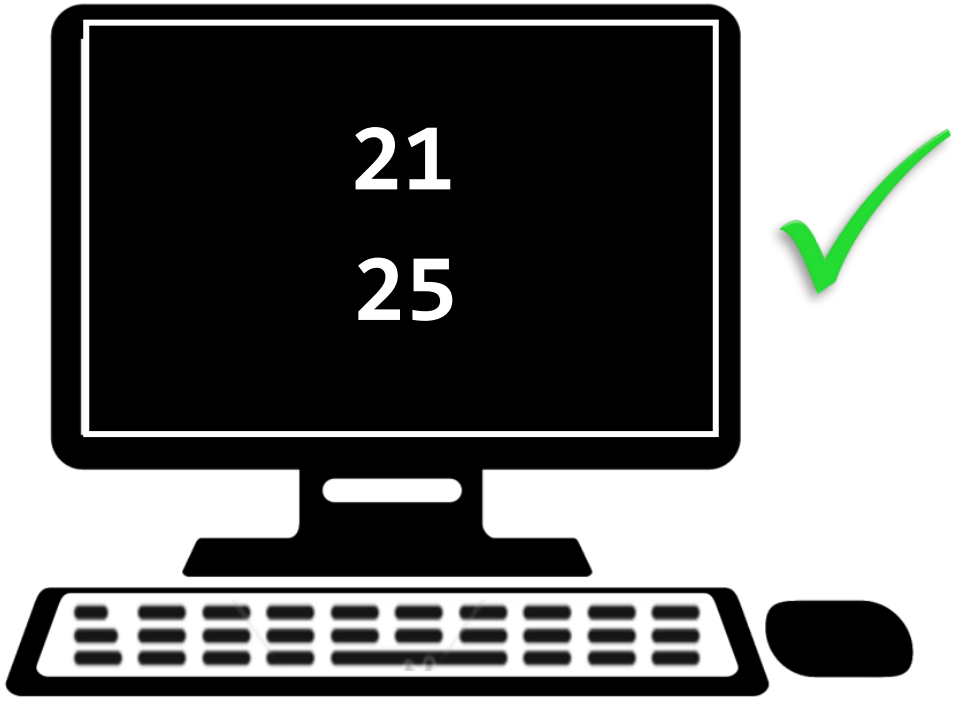
```
score3 = 56;
```

int Data Type (Ex.)

```
#include<iostream>
using namespace std;
int main()
{
    int age = 21;
    cout << age << endl;

    age = 25;
    cout << age << endl;

    return 0;
}
```



unsigned int

- unsigned int is used to store only **non-negative** whole numbers

  (0 and positive values).

```
#include<iostream>
using namespace std;
int main()
{
    unsigned int age = 21;

    int num1 =-48;

    unsigned int num2 = -13;

    return 0;
}
```

double & float Data Types

- The **double** and **float** data types are used for storing numbers with decimals.

  double provides higher precision compared to float.

```
#include<iostream>
using namespace std;
int main()
{
    double temperatureOfToday = 13.5;
    float price = 99.9;

    double quiz1;
    quiz1 = 4.7;

    return 0;
}
```

double & float Data Types

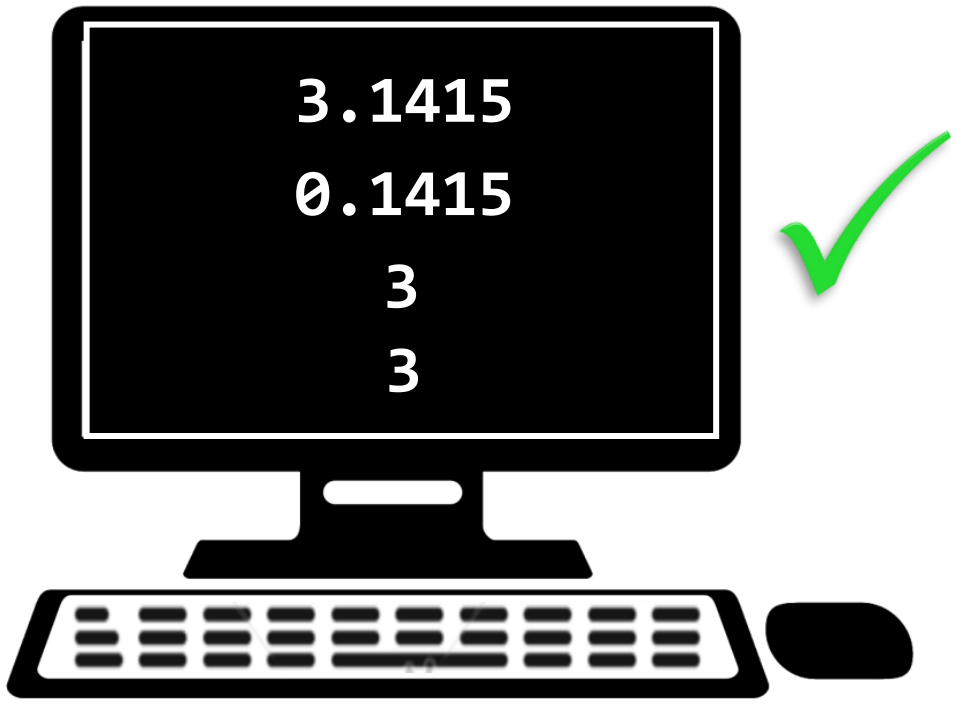
```
#include<iostream>
using namespace std;
int main()
{
    cout << 3.1415 << endl;

    cout << .1415 << endl;

    cout << 3.0 << endl;

    cout << 3. << endl;

    return 0;
}
```



Character (char) Data Type

- The char data type is used for storing single characters. Characters are enclosed in single quotes

```
#include<iostream>
using namespace std;
int main()
{
    char studentGrade = 'A';
    char x = '@';
    char num = '1';

    return 0;
}
```

string Data Type

- The string data type is used for storing sequences of characters (strings).

  Strings are enclosed in double quotes.

```
#include<iostream>
using namespace std;
int main()
{
    string courseName = "Programming I";

    string fullName;
    fullName = "Paywand";

    return 0;
}
```

boolean Data Type

- The **bool** data type is used for storing values with two states: **true** or **false**.

  It's often used in decision-making and control flow.

```
#include<iostream>
using namespace std;
int main()
{
    bool isTestPassed = true;
    bool evenNumber = false;

    return 0;
}
```

Activity #1

- Find errors in this code

```
#include<iostream>
using namespace std;
int main()
{
    int num = 3.14;

    char fullName = "Zhila";

    bool score = 75;

    return 0;
}
```

?!

Activity #2

- Data Type Selection

A. Storing GPA → double, float

B. Calculating price with tax → double, float

C. Tracking exam scores → int

D. Storing a single character → char

E. Storing Book Titles → string

F. Measuring Temperature → double, float

G. Counting Pages in a Book → int

H. Determining Pass/Fail Status → bool

Data Type Ranges

Data Type	Range
int	−2,147,483,648 to 2,147,483,647
unsigned int	0 to 4,294,967,295
float	~1.2E−38 to ~3.4E+38
double	~2.3E−308 to ~1.7E+308
char	−128 to 127 (or 0 to 255)

Overflow and Underflow

- **Overflow** occurs when a value exceeds the maximum representable value for a given data type.

```
#include<iostream>
using namespace std;
int main()
{
    int num1 = 2147483647;
    cout << num1 << endl;

    int num2 = 2147483648;
    cout << num2 << endl;

    return 0;
}
```

Overflow and Underflow

- **Underflow** occurs when a value becomes smaller than the minimum representable value for a given data type.

```
#include<iostream>
using namespace std;
int main()
{
    int num1 = -2147483648;
    cout << num1 << endl;

    int num2 = -2147483649;
    cout << num2 << endl;

    return 0;
}
```

Identifier (Variable Names)

- Regardless of which style you adopt, be consistent and make your variable names as sensible as possible. C++ is **case sensitive**.
- The first character must be one of the letters **a** through **z**, **A** through **Z**, or an underscore character (_).
- After the first character you may use the letters a through z or A through Z, the digits 0 through 9, or underscores.
- Uppercase and lowercase characters are **different**.

Identifier (Variable Names)

- Legal and Illegal Variable Names

Variable Name	Legal or Illegal
`dayOfWeek`	Legal.
`3dGraph`	Illegal. Variable names cannot begin with a digit.
`_employee_num`	Legal.
`June1997`	Legal.
`Mixture#3`	Illegal. Variable names may only use letters, digits, and underscores.

Identifier (Variable Names)

- C++ Key words can not be used as Identifier.

alignas	const	for	private	throw
alignof	constexpr	friend	protected	true
and	const_cast	goto	public	try
and_eq	continue	if	register	typedef
asm	decltype	inline	reinterpret_cast	typeid
auto	default	int	return	typename
bitand	delete	long	short	union
bitor	do	mutable	signed	unsigned
bool	double	namespace	sizeof	using
break	dynamic_cast	new	static	virtual
case	else	noexcept	static_assert	void
catch	enum	not	static_cast	volatile
char	explicit	not_eq	struct	wchar_t
char16_t	export	nullptr	switch	while
char32_t	extern	operator	template	xor
class	false	or	this	xor_eq
compl	float	or_eq	thread_local	

Variable Scope

- A variable must be declared before it is used in the program.
- If the variable used before declaring it, the compiler will generate an error.

```
#include<iostream>
using namespace std;
int main()
{
    cout << value;   //Error, value not defined yet!
    int value = 100;
    cout << value;   //No Error! inside scope

    return 0;
}
```

Constants

- Constants are values that cannot be modified after they are initialized in a program.

```
#include<iostream>
using namespace std;
int main()
{
    const float PI = 3.14;
    PI = 4.0;    // Error! value can not be changed

    return 0;
}
```

Data Types (Ex.)

```cpp
#include <iostream>
using namespace std;
int main() {
    // Student record example
    int studentId = 12345;
    string firstName = "Zhilwan";
    string lastName = "Ahmad";
    char group = 'A';
    double gpa = 3.85;
    bool isActive = true;

    // Course details
    string courseName = "Programming I";
    int totalStudents = 30;
    double passingGrade = 60.0;

    return 0;
}
```

Activities and Next Lecture's Topic

Activities

- Review this lecture note
- Practice

Next Lecture's Topic

- Expressions and Interactivity

References

- Gaddis, T. (2014). Starting out with C++: Early objects (7th ed.). Pearson Education.

Thank You!