

# CS 32 Intro to CS II

DIS 1B, Week 1, Fall 2025

Zheng Luo

Slides & Code available at "<https://tinyurl.com/cs32-1b-w1>"

# Agenda

- ▶ Installation of Visual Studio Code & C++ Compiler
- ▶ Function Overloading
- ▶ Constructor Overloading
- ▶ Pointers & Address

## VS Code & GCC

- ▶ Download and install Visual Studio Code from “<https://tinyurl.com/cs32vscode>”
- ▶ For Windows, “<https://tinyurl.com/cs32vscodeWin>”
- ▶ For Mac / Linux, “<https://tinyurl.com/cs32vscodeMac>”

# GCC on Windows

For Windows:

1. Go to <https://www.msys2.org/>, download and install GCC
2. Press Win + q and search for "MSYS2 UCRT64"
3. Enter the following 4 commands
  - ▶ `pacman -S mingw-w64-ucrt-x86_64-gcc`
  - ▶ `pacman -S mingw-w64-ucrt-x86_64-gdb`
  - ▶ `pacman -S mingw-w64-ucrt-x86_64-mesa`
  - ▶ `pacman -S mingw-w64-ucrt-x86_64-freeglut`
4. WIN + q → search for "Advanced System Settings"
5. Open environment variables
6. Enter  
"C:\msys64\ucrt64\bin"
7. Click Ok
8. WIN + q → search for "cmd"
9. Enter "gcc --version" to confirm installation

# Function Overloading

Create multiple **functions** of the **same name** with **different implementations**  
(funcOverloading.cpp from from "<https://tinyurl.com/cs32-1b-w1>")

```
1  int add(int a, int b) {return a + b;} // a
2
3  double add(double a, double b) {return a + b;} // b
4
5  double add(int a, double b) {return a + b;} // c
6
7  int add(int a, int b, int c) {return a + b + c;} // d
```

# Function Overloading

Create multiple **functions** of the **same name** with **different implementations**  
(funcOverloading.cpp from from "https://tinyurl.com/cs32-1b-w1")

```
1  int add(int a, int b) {return a + b;} // a
2
3  double add(double a, double b) {return a + b;} // b
4
5  double add(int a, double b) {return a + b;} // c
6
7  int add(int a, int b, int c) {return a + b + c;} // d
```

What sets a function apart from others?

# Function Overloading

Create multiple **functions** of the **same name** with **different implementations**  
(funcOverloading.cpp from from "https://tinyurl.com/cs32-1b-w1")

```
1  int add(int a, int b) {return a + b;} // a
2
3  double add(double a, double b) {return a + b;} // b
4
5  double add(int a, double b) {return a + b;} // c
6
7  int add(int a, int b, int c) {return a + b + c;} // d
```

What sets a function apart from others? **Function Signature**

# Function Overloading

Create multiple **functions** of the **same name** with **different implementations**  
(funcOverloading.cpp from from "https://tinyurl.com/cs32-1b-w1")

```
1  int add(int a, int b) {return a + b;} // a
2
3  double add(double a, double b) {return a + b;} // b
4
5  double add(int a, double b) {return a + b;} // c
6
7  int add(int a, int b, int c) {return a + b + c;} // d
```

What sets a function apart from others? **name + parameter list** - return type



# Object

is a unit of data (properties) and actions (methods)

# Object

is a unit of data (properties) and actions (methods)

Examples: BruinCard, Broad Art Center and UCLA are objects

# Object

is a unit of data (properties) and actions (methods)

Examples: BruinCard, Broad Art Center and UCLA are objects

An object in C++ is a concrete instance of a class.

# Object

is a unit of data (properties) and actions (methods)

Examples: BruinCard, Broad Art Center and UCLA are objects

An object in C++ is a concrete instance of a class.

So we need constructor(s) to construct the object.

# Object

is a unit of data (properties) and actions (methods)

Examples: BruinCard, Broad Art Center and UCLA are objects

An object in C++ is a concrete instance of a class.

So we need constructor(s) to construct the object.

```
1 class BruinCard {  
2   public :  
3       BruinCard(int id , string name) {...}  
4  
5       BruinCard(int id , string name, double balance) {...}  
6   };
```

# Overloaded Constructors

(constructorOverloading.cpp from "<https://tinyurl.com/cs32-1b-w1>" )

```
1 class BruinCard {  
2 public :  
3     BruinCard(int id , string name) {...}  
4  
5     BruinCard(int id , string name, double balance) {...}  
6 };
```

# Address & Pointer

## Real-world Example 1

- ▶ Object: Broad Art Center (the entire building)
- ▶ Address: "240 Charles E Young Dr N, Los Angeles, CA 90095"
- ▶ Pointer: a note with this address written on it

# Address & Pointer

## Real-world Example 1

- ▶ Object: Broad Art Center (the entire building)
- ▶ Address: "240 Charles E Young Dr N, Los Angeles, CA 90095"
- ▶ Pointer: a note with this address written on it

How did you find your way here today?



# Address & Pointer

## Real-world Example 1

- ▶ Object: Broad Art Center (the entire building)
- ▶ Address: "240 Charles E Young Dr N, Los Angeles, CA 90095"
- ▶ **Pointer**: a note with this address written on it

How did you find your way here today?

# Address & Pointer

## Real-world Example 2

- ▶ Object: My BruinCard
- ▶ Address: "In my wallet on the podium"
- ▶ Pointer: a note with this address written on it

# Address & Pointer

## Real-world Example 2

- ▶ Object: My BruinCard
- ▶ Address: "In my wallet on the podium"
- ▶ Pointer: a note with this address written on it

What do we need to get my wallet?

# Address & Pointer

## Real-world Example 2

- ▶ Object: My BruinCard
- ▶ Address: "In my wallet on the podium"
- ▶ **Pointer**: a note with this address written on it

What do we need to get my wallet?

# Address & Pointer

## C++ Example

### ► Object:

```
BruinCard myCard(1, "Z", 100.0);
```

### ► Address:

```
0x0000000016fdfeca0
```

### ► Pointer:

```
BruinCard* myCardPtr = &myCard;
```

# Address & Pointer

## C++ Example

### ► Object:

```
BruinCard myCard(1, "Z", 100.0);
```

### ► Address:

```
0x000000016fdfeca0
```

### ► Pointer:

```
BruinCard* myCardPtr = &myCard;
```

What do we need to find the object *myCard*?

# Address & Pointer

## C++ Example

### ► Object:

```
BruinCard myCard(1, "Z", 100.0);
```

### ► Address:

```
0x000000016fdfeca0
```

### ► Pointer:

```
BruinCard* myCardPtr = &myCard;
```

What do we need to find the object *myCard*?

## Address & Pointer

(addNptr.cpp / addNptrVerbose.cpp from "<https://tinyurl.com/cs32-1b-w1>" )

```
1    void printBalance(BruinCard card){
2        cout << card._balance << endl;
3    }
4
5    void printBalance(BruinCard* cardPtr){
6        cout << cardPtr->_balance << endl;
7    }
8
9    void printBalanceByAdd(BruinCard& card){
10        cout << card._balance << endl;
11    }
```