Passing by reference Vs by Value

what is "reference"?

it's the address of your variable in the memory

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

int main()
{
    int x = 0; // our variable

    cout << "value of x = " << x << endl; // print the value

    cout << "address of x is: " << &x; // print the address

    return 0;
}
```

```
value of x = 0
address of x is: 0x7fff5d72f994

...Program finished with exit code 0
Press ENTER to exit console.
```

Passing by Value

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

int add(int x) // pass x by value
{
    x +=1;
    return x;
}
int main()
{
    int x = 0; // our variable
    add (x); // call the function
    cout << "x = " << x;

    return 0;
}
```

```
x = 0

...Program finished with exit code 0
Press ENTER to exit console.
```

Notice that the x was 0 even after we called the function that changes x to x+1

That's because x was changed just in the function not in the memory.

Passing by reference

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

int add(int &x) // pass x by reference
{
    x +=1;
    return x;
}
int main()
{
    int x = 0; // our variable
    add (x); // call the function
    cout << "x = " << x;

    return 0;
}
```

```
x = 1

...Program finished with exit code 0
Press ENTER to exit console.
```

Example: withdrawing money from your bank account

```cpp
using namespace std;
int withdraw(double b, double a) // pass b (the balance) by value
{
    b = b - a; // balance = balance - amount we want to withdraw
    return b;
}
int main()
{
    double balance = 200;
    withdraw(balance, 100);
    cout << "balance = " << balance;
    return 0;
}
```

input

```
balance = 200

...Program finished with exit code 0
Press ENTER to exit console.
```

Still 200 ←

Now you know that the balance didn't change because we passed it by value

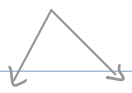
Correct code:

```cpp
using namespace std;
int withdraw(double& b, double a) // pass b (the balance) by reference
{
    b = b - a; // balance = balance - amount we want to withdraw
    return b;
}
int main()
{
    double balance = 200;
    withdraw(balance, 100);
    cout << "balance = " << balance;
    return 0;
}
```

input

```
balance = 100

...Program finished with exit code 0
Press ENTER to exit console.
```

Note: Types of functions



return value — int, str, bool ...

No return value — void

Classes

header file (.h) syntax

```
Class className
{
    // methods
    public:
        dataType methodName(parameterType);
                                  // if you have any
    // data members
    private:
        dataType variableName;
};
```

.cc file (the implementation) Syntax

```
#include "className.h"
// write the body of your methods
type className::methodName(parameterType parameterName)
{
    // body
}
```

Note: setters are void, and they take parameters

getters have a return type, and the don't take parameters

(return type depends on what are you getting/returning (which data member))

testDriver.cc syntax

```
#include "ClassName.h"
int main()
{
    // declaring objects
    ClassName  objectName;

    // Using the class's methods
    objectName.methodName();
                    add the parameters if you have any
    return 0;
}
```

Student class example:

Student.h

C: > Users > reemH > OneDrive > Desktop > Uni > Sophon

```h
#include <string>
class Student
{
    public:
        //prototypes
        void setName(string);
        string getName();
    private:
        string studentName;
        char studentId;
        int studentAge;
};
```

Student.cc

C: > Users > reemH > OneDrive > Desktop > Uni > Soph

```cpp
#include "Student.h"

void Student::setName(string name)
{
    studentName = name;
}
string Student::getName()
{
    return studentName;
}
```

StudentTestDriver.cc

C: > Users > reemH > OneDrive > Desktop > Uni > Sophon

```cpp
#include <iostream>
#include "Student.h"
#include <string>
using namespace std;

int main()
{
    Student s;
    s.setName("Reem");
    cout << s.getNAme();
    return 0;
}
//output: Reem
```

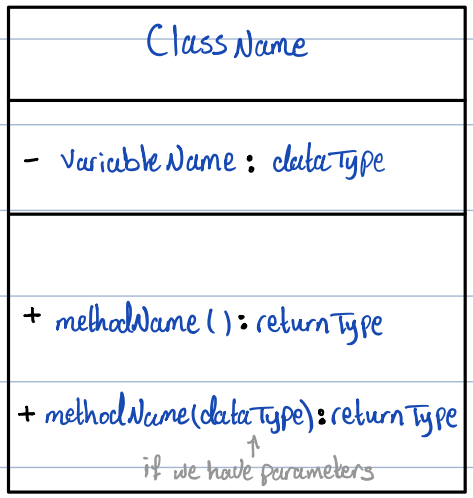
You can take the parameter as an input

StudentTestDriver.cc

C: > Users > reemH > OneDrive > Desktop > Uni > Sophomore > Sophomore1 > CS1131_Advanced Progra

```cpp
#include <iostream>
#include "Student.h"
#include <string>
using namespace std;

int main()
{
    Student s;
    string name;

    cout << "Enter name: ";
    cin >> name;

    s.setName(name);
    cout << s.getName();

    return 0;
}
//the output will be the same name you entered

```

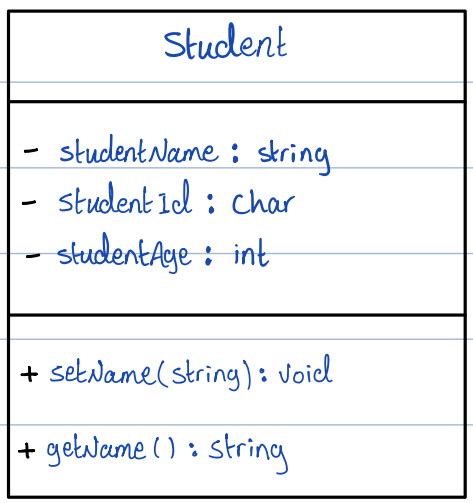
UML Diagram



Private: - data members / fields
Public: + methods

---

student class example:



Student.h

C: > Users > reemH > OneDrive > Desktop > Uni

```
#include <string>
class Student
{
    public:
        //prototypes
        void setName(string);
        string getName();
    private:
        string studentName;
        char studentId;
        int studentAge;
};
```