7COM1025
Programming for Software Engineers
Lecture 9

Dr. Renato C. de Amorim

CALL BY VALUE/CALL BY REFERENCE

```
#include <iostream>
using namespace std;
double reciprocal(double x);
int main()
{
    double t = 10.0;
    cout << "Reciprocal of 10.0 is 
"<<reciprocal(t)<<endl;
    cout<<"Value of t is still: " <<t<<endl;
    return 0;
}

double reciprocal(double x)
{
    x=1/x;
    return x;
}
```

```
#include <iostream>
using namespace std;
void swap(int *x, int *y);
int main()
{
    int a=10, b=5;
    cout<<"a: "<<a<<"b: "<<b<<endl;
    swap(&a, &b);
    cout<<"a: "<<a<<"b: "<<b<<endl;
    return 0;
}

void swap(int *x, int *y)
{
    int temp = *x;
    *x=*y;
    *y=temp;
}
```

AUTOMATIC CALL BY REFERENCE

You can tell C++ to pass by reference rather than pass by value.

```
#include <iostream>
using namespace std;
void swap(int &x, int &y);
int main()
{
    int a=10, b=5;
    cout<<"a: "<<a<<" b: "<<b<<endl;
    swap(a, b);
    cout<<"a: "<<a<<" b: "<<b<<endl;
    return 0;
}
void swap(int &x, int &y)
{
    int temp = x;
    x=y;
    y=temp;
}
```

PROBLEM 9.1

Write a program allowing the user to enter an unknown number of integers and then provide the average of these numbers. Your program should have a function to calculate this average.

RETURNING REFERENCES

```
#include <iostream>
using namespace std;
double &f();
double val=100.0;
int main()
{
    double x;
    cout<<f()<<endl;
    x=f();
    cout<<x<<endl;
    f() = 99.1;
    cout<<f()<<endl;
    return 0;
}
double &f()
{
    return val;
}
```

University of Hertfordshire UH

THE ONE HUNDRED UNDER FIFTY

RETURNING REFERENCES

```
#include <iostream>
using namespace std;
double &change_it(int i);
double vals[]={1.1,2.2,3.3,4.4,5.5};
int main()
{
    int i;
    cout<<"Original values: ";
    for (i=0; i<5;i++)
        cout<<vals[i]<< ' ';
    cout<<endl;
    change_it(1)=5298.23;
    change_it(3)=-98.8;
    cout<<"Values: ";
    for (i=0; i<5;i++)
        cout<<vals[i]<< ' ';
    cout<<endl;
    return 0;
}
double &change_it(int i)
{
    return vals[i];
}
```

You can't return a reference to a local variable

```
int &f()
{
int i=10
return i;
}
```

INDEPENDENT REFERENCES

That’s a stand alone reference variable.

Non-parameter reference variables are seldom used.

```
#include <iostream>
using namespace std;
int main()
{
   int j,k;
   int &i=j;
   j=10;
   cout<<j<<" "<<i<<endl;
   k=121;
   i=k;
   cout<<j<<endl;
   return 0;
}
```

POINTERS VS REFERENCES

Restrictions to references:

- You can’t reference a reference variable.
- You can’t create arrays of references.
- You cannot create a pointer to a reference (ie. you can’t apply the & operator to a reference.)
- You can’t get the address of a reference like you can with pointers.

A pointer and a reference occupy the same amount of memory

FUNCTION OVERLOADING

Two of more functions can share the same name as long as their parameter list is different.

The type and/or the number of parameters of each overloaded function must differ.

The difference cannot be <u>only</u> their return type.

FUNCTION OVERLOADING

```
#include <iostream>
using namespace std;
void f(int x);
void f(double x);
int main()
{
    int a=10;
    double d=123.123;
    f(a);
    f(d);
return 0;
}
void f(int x)
{
    cout<<x<<" is int."<<endl;
}
void f(double x)
{
    cout<<x<<" is double"<<endl;
}
```

What if you had the following inside main()?

```
short s=99;
float r=11.5F;
f(s);
f(r);
```

Int and double!

PROBLEM 9.2

By creating your own IO functions you can tailor it to your own needs.

1- create a “input” function that allows a text. For instance, instead of:

```
cout<<“Question: “;
cin>>var;
```

You do:

```
input(“Question”, var);
```

Create two functions “print” and println” so that

```
print(1);
println(‘X’)
print(“Any text. “);
print(123.123);
```

Displays

```
1X
Any text. 123.123
```

DEFAULT FUNCTION ARGUMENTS

```
#include <iostream>
using namespace std;
void myfunc(int x=0, int y=100);
int main()
{
    myfunc(1,2);
    myfunc(10);
    myfunc();
    return 0;
}
void myfunc(int x, int y)
{
    cout<<"x: "<<x<<" and y: "<<y<<endl;
}
```