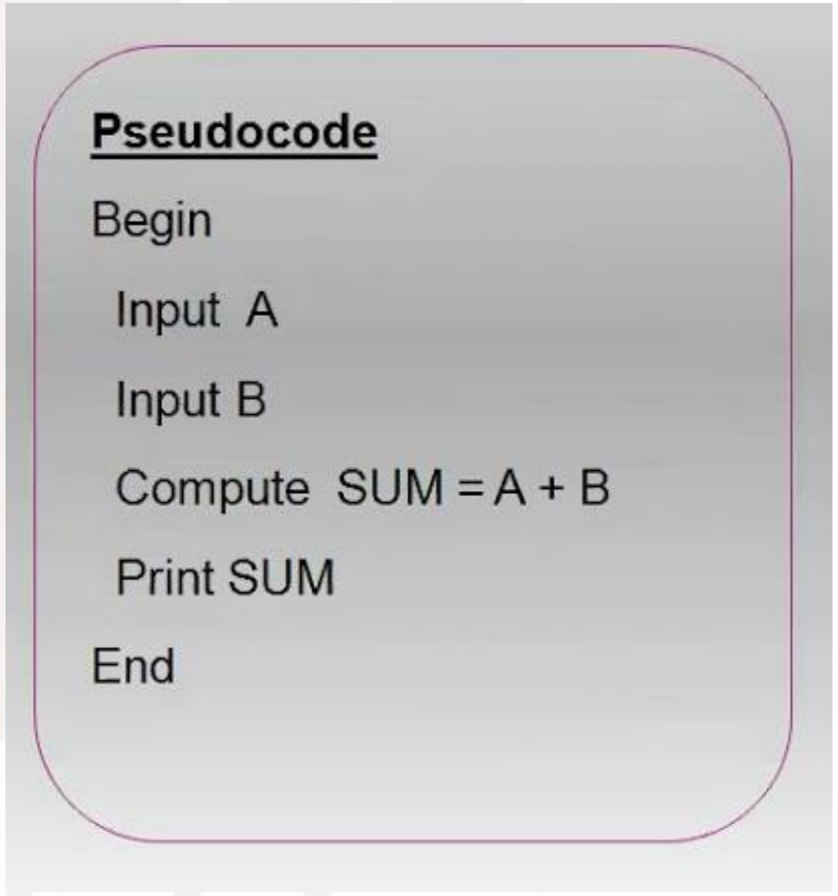
CIS 129
Advanced Computer Programming

Chapter 4: User Defined Functions

Mr. Horence Chan

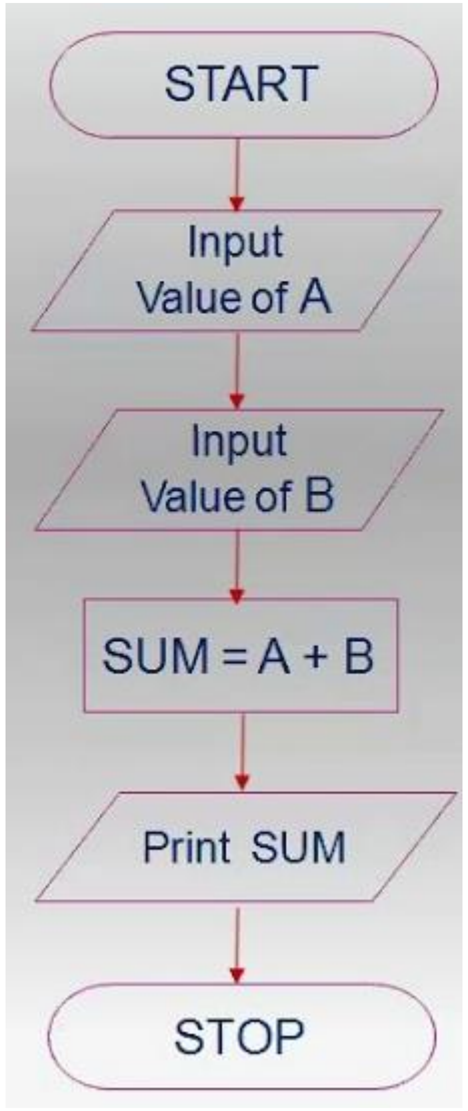
Pseudocode & Flow Chart

- Before writing the program directly, it is a good practice to write the "**pseudocode**" first
- Using plain English to describe what's supposed to happen, then keep expanding each sentence until it's sufficiently detailed that you can express it as if-statements, loops, etc.
- Often some of the initial English descriptions will describe good ways to divide up the code into functions.



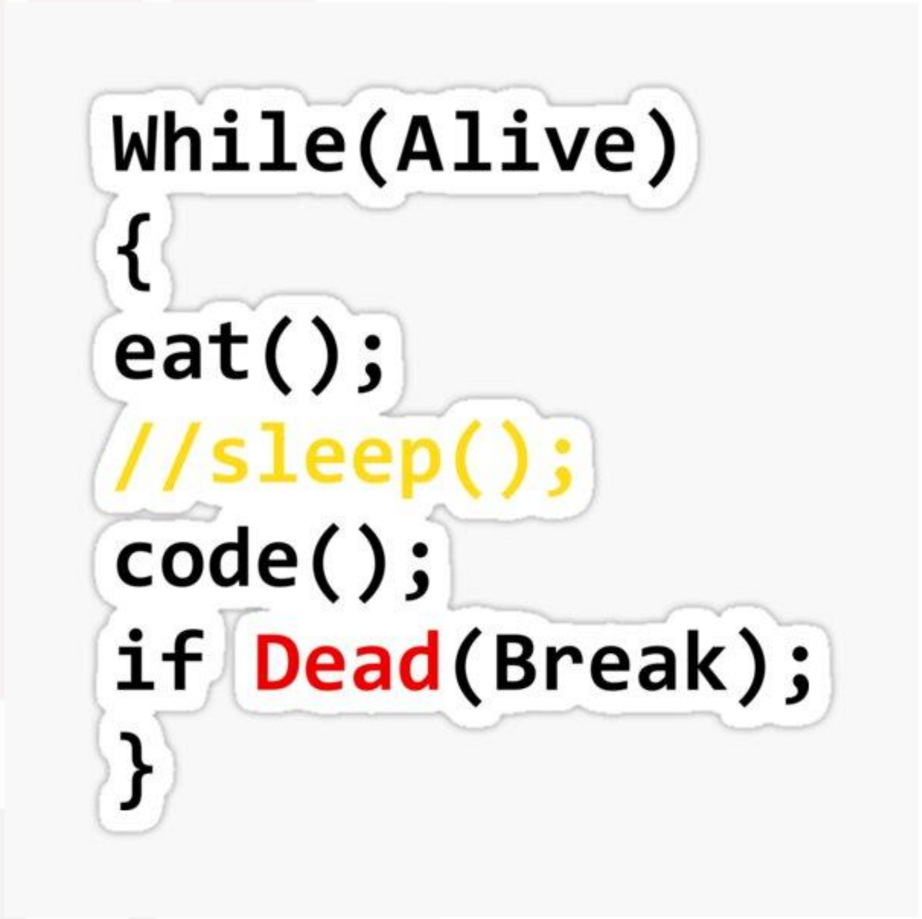
Pseudocode

```
Begin
  Input  A
  Input B
  Compute  SUM = A + B
  Print SUM
End
```



Function

- Dumping all the code into `main` would be extremely long and difficult to keep track of.
- Nobody who read a single line would have a clue where that line fit in. We would lose track of our programming goals.
- It would be much more intuitive to break the code in serval parts, which is call ______________
- A function is a block of code with a name



```
While(Alive)
{
eat();
//sleep();
code();
if Dead(Break);
}
```

Function

- For example, we are trying to program a robot to launch someone from Hong Kong to Tokyo via rocket, here is the pseudocode:

```
int main() {
    buildRocket();
    setUpRocket();
    fireRocket();
}
```

- This style is often a good design for main – main a few calls to some functions that do all the real work.
- Each of the functions `buildRocket()`, `setUpRocket()`, `fireRocket()` is said to be "invoked" or "called" via a "function call" from "calling function" or "caller" (in this case, main).
- To call a function, type the ________ of the function, followed by _______________.

Predefined functions

- Example of predefined functions in C++ libraries:

Function	Purpose	Parameter(s) Type /Result	Example
`floor(x)`	Returns the largest whole number that is not greater than x	`double`	`floor(45.67) =` ________
`islower(x)`	Return `true` if `x` is a lowercase letter; otherwise, it returns `false`	`int`	`islower('h')` is ________
`isupper(x)`	Return `true` if `x` is a uppercase letter; otherwise, it returns `false`	`int`	`isupper('K')` is ________
`pow(x, y)`	Return x^y; if `x` is negative, `y` must be a whole number	`double`	`pow(0.16 , 0.5) =` ________
`sqrt(x)`	Returns the nonnegative square root of `x`; `x` must be nonnegative	`double`	`sqrt(4.0) =` ________

User defined functions (without return value)

```
#include <iostream>
using namespace std;
_________________________                // ___________ the function
int main()
{
     __________________                  // ___________ the function
      return 0;
}
____________________________  {
     cout << "Hello world!";             // ___________ of the function
}
```

This definition specifies that we want to name the sequence of commands within the curly braces {…} ____________________, so that we can then call it from another function, such as main, with the syntax _____________________

User defined functions (without return value)

```
#include <iostream>
using namespace std;
____________________
int main()
{
    _______________
     return 0;
}
_______________________  {
    cout << "Hello world!";
}
```

- The _____________ *return type* specifies that there is _____________________, which generally means that this function is for issuing instructions, not asking a question.
- Returning a value from a ________ function is a syntax error.
- Not returning a value from a non-void function is not a syntax error but sometimes may cause runtime errors.

User defined functions (With return value)

```
#include <iostream>
using namespace std;
bool ______________( int x, int y);
int main()
{
    cout << ______________(6,2) << endl;
    cout << ______________(7,5) << endl;
}
bool ______________(int x, int y) {
    if (x % y == 0)
        return true;
    else
        return false;
}
```

- In this example, we are asking “Is x a multiple of y?”
- `bool`: ___________
- `isMultiple`: Function name
- `x` and `y` are ________________
- `return true` and `return false`: ____________________

Output

User defined functions (With return value)

```
#include <iostream>
using namespace std;
int big(int a, int b);
int main(void) {
    int bigger;
    bigger = big(31, 24);
    cout << bigger << " is bigger!";
    cout << endl;
    return 0;
}
int big(int a, int b) {
    if (a > b) {
        __________
    }
    else {
        __________
    }
}
```

Output

int main()
I think I forgot something.
void main()
If you forgot, then it wasn't important.
Yeah, you're right.
return 0;

- In “with return value” function, remember to write code(s) contains ______________!!!

Scope

```
#include <iostream>
using namespace std;
const int global = 1;
int main()
{
    int local = 0;
    return 0;
}
```

- Variables exist within ***scopes*** – blocks of code within which identifiers are valid. An identifier can be referenced anywhere within its scope, if the reference comes after its declaration.
- ______________________ – variables declared outside of any function – have file scope, meaning they can be referred to from _______________ in the file. Global variables should generally be avoided, except for global named ______________________.

Scope

```
#include <iostream>
using namespace std;
const int global = 1;
int main()
{
    int local = 0;
    return 0;
}
```

- _____________________ - the set of braces in which a variable was declared ends, the variable goes out of scope, i.e. it can no longer be referenced as an identifier. The program usually ____________ variables that have gone out of scope from memory. The scope of arguments to a function is the entire function body.

Reference

```
#include <iostream>
using namespace std;
void reference(int _____ x, int y);
int main()
{
    int a = 0;
    int b = 0;
    reference(a, b);
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;
    return 0;
}
void reference(int _____ x , int y)
{
    x = 2;
    y = 3;
    cout << "x = " << x << endl;
    cout << "y = " << y << endl;
}
```

- A reference (______) is an alias for another variable
- If the value of the reference is ___________, the value of another variable also ___________

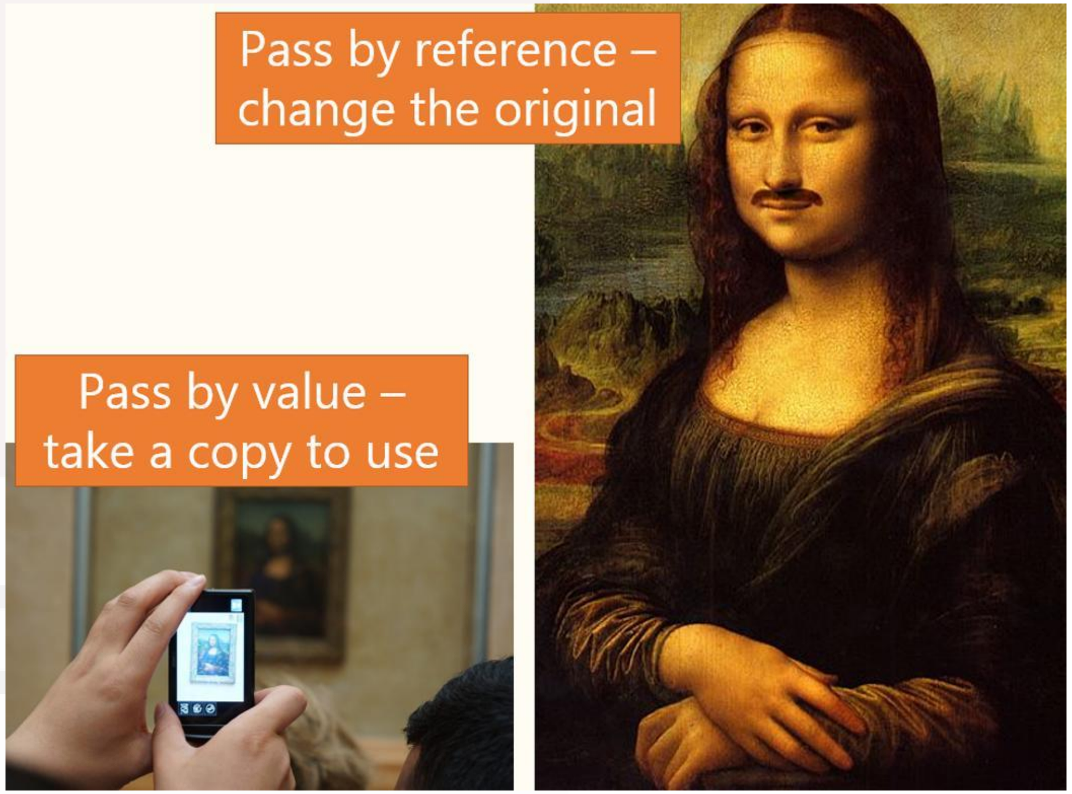
Output

```
x = 2
y = 3
a = ______
b = ______
```

Reference

```
#include <iostream>
using namespace std;
void reference(int _____ x, int y);
int main()
{
    int a = 0;
    int b = 0;
    reference(a, b);
    cout << "a = " << a << endl;
    cout << "b = " << b << endl;
    return 0;
}
void reference(int _____ x , int y)
{
    x = 2;
    y = 3;
    cout << "x = " << x << endl;
    cout << "y = " << y << endl;
}
```

- `x` and `a` : "pass by ____________"
- `y` and `b` : "pass by ____________"


Pass by reference – change the original
Pass by value – take a copy to use

Conversion

```
#include <iostream>
using namespace std;
int main (){
    float x = 67.89;
    int y;
    y = static_cast < __ >(x);
    cout << "x = " << x << endl;
    cout << "y = " << y << endl;
    return 0;
}
```

- `static_cast <>()` is used to convert the data type of a variable
- <>: input the new ____________
- (): input the ___________ need to convert

Output:

```
x = 67.89
y = 67
```

Function and File Input/Output

```
void ReadandWrite(ifstream _____inp, ofstream _____out, string food, int price);
int main(){
       ifstream inFile;
       ofstream outFile;
       string inputFile, product;
       int prices;
       cout << "Enter the file name: ";
       cin >> inputFile;
       cout << endl;
       inFile.open(inputFile___________);
       outFile.open("price_output.out");
       ReadandWrite(inFile, outFile, product, prices);
       inFile.close();
       outFile.close();
    return 0;
}
```

- If the file name is input by the user,
- Use "_____________" to open the file

- When passing ___________________ and ___________________ datatype to a function, they must be ___________________

Function and File Input/Output

```
void ReadandWrite(ifstream _____inp, ofstream _____out, string food, int price){
    inp >> food >> price;
    while(_____){ // While inp is _________ (i.e. still _____________from the file)
        out << "The price of " << food << " is $" << price << "." << endl;
        inp >> food >> price;
    }
}
```

- Input file

```
burger      15
fries       11
ice-cream   9
coke        7
Steak       100
```

- Output file

```
The price of burger is $15.
The price of fries is $11.
The price of ice-cream is $9.
The price of coke is $7.
The price of steak is $100.
```