CISC 1600/1610
Computer Science I

Functions, continued

Professor Daniel Leeds
dleeds@fordham.edu
JMH 328A

Variable scope

Variables declared in a function
• are local to that function
• are invisible to all other functions

int main() is a function

```
int newFunc(int a);

int main() {
    int a=5, b, c=5;
    b = newFunc(a);
    cout << a << " " << b << " " << c << endl;
    return 0;
}

int newFunc(int a) {
    int c=12;
    return a*5+c;
}
```

What does this code do?

```
float triple(float inNum) {
    float tripledNum;
    tripledNum=3*inNum;
    return tripledNum;
}
```

```
int main() {
    int a=5, b, c=5;
    b = newFunc(a);
    cout << a << " " << b << " " << c << endl;
    return 0;
}
```

Formal parameters

“Formal parameters” are the variables in the function head

```
float triple(float inNum) {
    float tripledNum;
    tripledNum=3*inNum;
    return tripledNum;
}
```

Formal parameter names

• Parameter names do not have to match names of variables used in function call
• Different programmer can write int main() and functions used by int main()
Broader scope: global variables

- Global variables visible to all functions
- Declared outside of all functions
- Must be declared prior to first use

```cpp
#include<iostream>
using namespace std;
const float PI=3.14;
// visible to main and to areaCircle
// compute area of circle
float areaCircle(float radius);
int main() { ...}
float areaCircle(float radius) {...}
```

More on global variables

- Useful to define global constants
- Very risky to define non-constant global variables
  -- try to keep track of what functions change the variable

void functions

- void function returns no value

Example definition:
```cpp
void greetUser(string userName){
    cout << "Hello " << userName
    << endl;
    return;
}
```

Example call:
```cpp
greetUser(userName);
```

Use of return;

- In void function, can use return;
- When evaluated, return; terminates function

```cpp
NOT: cout << greetUser(userName);
```