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

What does this code do?

```
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;
}
```


Formal parameters

"Formal parameters" are the variables in the function head

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

Function head

Function body

Formal parameters

- **Local** to the function
- Used as if they were declared in function body – **do not** re-declare in function body
- When function is called, parameters initialized to the values of the arguments in the function call

```
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

```
#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) {...}
```

7

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

8

`void` functions

- void function returns no value

Example definition:

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

Example call:

```
  greetUser(userName);
```

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

11

Use of `return;`

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

12