Lecture 17

Devendra Ghate

05/01/2021

Code never lies, comments sometimes do.
Ron Jeffries

There are only two types of languages: The ones people complain about, and the ones nobody uses.
Bjarne Stroustrup

Functions

- Our programs are getting bigger.
- We might need to **reuse** a portion of code multiple times.
- Our expertise is improving. We might want to **share** our expertise with others
- We create functions to better **manage** our code and distribute it widely (./31-functions.cpp)
- We also create them to **limit access** to our algorithms and data

Functions

- Have an input and an output.
- Datatype of both has to be defined.
- Might not have output if type is `void`.

Passing variable by value

```
 ...
double cube(double x);
int main()
{

   ...
   double side = 5;
   double volume = cube(side);
   ...
}

double cube(double x)
{
 return x * x * x;
}
```

creates variable called `side` and assigns it the value 5

original value

passes the value 5 to the `cube( )` function

creates variable called `x` and assigns it passed value 5

copied value

Passing variable by value

- Scope of variables (./32-variableScope.cpp)
- Only the value of a variable is passed from the calling function to the called function.
- All the variables created inside a function are lost once function exits.
- If the same function is called again, all the local variables are created and initialised once again.

Passing multiple inputs

- `double add(double a, double b)`
  - Variables to be separated by comma
- `double add(double a, b)` not allowed

Recursion

- Self referencing functions
- Not allowed for `main`
- Example: (./34-recursion.cpp)

`inline` functions

```
inline <output-type> functionName(<input-type> input){
    .
    .
    .
}
```

- No need to declare a prototype
- Recommended way of declaring functions instead of the macro `#define`

Creating a library

- Example: (./functions/*)
- Prototype functions have to declared in the header files (`encrypt.h`)
  - Tells the compiler about the input and output datatype of the functions.

Q: Why doesn't compiler look at the function source to figure out the I/O format?

- Function itself can be defined in a separate source file (`encrypt.cpp`)
- Compiler can be used to compile functions separately (`encrypt.o`) and then linked to create a library (`encryption.a`)

Creating a library

```
# Creates object files of functions
g++ -c encrypt.cpp decrypt.cpp
# Agregating all function object files to create library
ar -crv encryption.a encrypt.o decrypt.o
# Libraries can then be linked to the main function
g++ -I ./ -o libraryMain.out main.o encryption.a
```