

Overloading

Understand and remember.

- More than syntactic sugar.
- This is how a lot of stuff works under the hood (e.g. inheritance)

Function overloading - C

```
#include <stdio.h>
void foo()
{
    printf ("foo()\n");
}
void foo(int n)
{
    printf ("foo(%d)\n", n);
}
int main()
{
    foo(12);
    foo();
    return 0;
}
```

Compilation output:

**Error:
Multiple
definition of foo**

Function overloading – C++

```
#include <iostream>
void foo() {
    std::cout << "foo()\n";
}
void foo(int n) {
    std::cout<<"foo("<<n<<")\n";
}
int main() {
    foo(12);
    foo();
}
```

Output:

Compile, and print:

```
foo(12)
foo()
```

Default parameters

```
#include <iostream>
```

```
void foo(int n=5)  
{  
    std::cout << n;  
}
```

```
int main()  
{  
    foo();  
}
```

Output:

Compile, and print:
foo(5)

Overload resolution

1. Find all functions with same name “candidates”. Let’s call them O1.
2. Find O2 subset of O1 which have the correct number of arguments - “viable candidates”
3. Find O3 subset of O2 with best matching arguments.
if $|O3|=1$
 use that function.
else (0 or more than 1):
 emit compilation error.

Overload – Examples (folder 4)

- power functions;
- How does it work?
- Try the following in <http://godbolt.org> , first in C mode and then in C++ mode:

```
#include <math.h> /* log,exp */
```

```
int power(int a, unsigned int b) {  
    return b==0? 1: a*power(a,b-1);  
}
```

```
double power(double a, double b) {  
    return exp(b*log(a));  
}
```