601.220 Intermediate Programming

Function overloading

Function overloading

C++ compiler can distinguish functions with same name but different parameters

```cpp
// overload1.cpp:
#include <iostream>

using std::cout;  using std::endl;

void output_type(int)   { cout << "int" << endl; }
void output_type(float) { cout << "float" << endl; }

int main() {
    output_type(1);   // int argument
    output_type(1.f); // float argument
    return 0;
}
```

Function overloading

```
$ g++ -c overload1.cpp -std=c++11 -pedantic -Wall -Wextra
$ g++ -o overload1 overload1.o
$ ./overload1
int
float
```

Function overloading

But it *cannot* distinguish functions with same name & parameters but different return types

```
// overload2.cpp:
#include <iostream>

using std::cout;  using std::endl;

int   get_one() { return 1; }
float get_one() { return 1.0f; }

int main() {
    int i = get_one();
    float f = get_one();
    cout << i << ' ' << f << endl;
    return 0;
}
```

Function overloading

```
$ g++ -c overload2.cpp -std=c++11 -pedantic -Wall -Wextra
overload2.cpp:6:7: error: ambiguating new declaration of ‘float get_one()’
    6 | float get_one() { return 1.0f; }
      |       ^~~~~~~
overload2.cpp:5:7: note: old declaration ‘int get_one()’
    5 | int   get_one() { return 1; }
      |       ^~~~~~~
```

Quiz!

What output is printed by the following code?

```
#include <iostream>

char f(int c) {
  if (c % 2 == 0) { return 'X'; }
  else            { return 'Y'; }
}

int f(char c) {
  return (c - '0') * 11;
}

int main() {
  std::cout << f('7') << ","
            << f(7) << std::endl;
  return 0;
}
```

A. `77,X`

B. `77,Y`

C. `X,77`

D. `Y,77`

E. The code does not compile