COMP6771 Week 8.2

Template Metaprogramming

Metaprogramming

- **Metaprogramming**: Writing of computer programs with the ability to treat other program code as their data.
- In C++ we can do metaprogramming via either:
  - Use of constexpr
  - Template metaprogramming
- **C++ Template Metaprogramming**: Compile time calculated values created through the recursive instantiation of templates

Template Metaprogramming

constexpr vs template meta-programming

```cpp
#include <iostream>

constexpr int factorial (int n) {
  return n > 0 ? n * factorial( n - 1 ) : 1;
}

int main() {
  std::cout << factorial(6) << std::endl;
}
```

```cpp
#include <iostream>

template<int n> struct Factorial {
  static const long val = Factorial<n-1>::val * n;
};

template<> struct Factorial<0> {
  static const long val = 1; // must be a compile-time constant
};

int main() {
  std::cout << Factorial<6>::val << std::endl;
}
```

Template Metaprogramming

An over-the-top example

```
#include <iostream>

template<int I, int J>
struct IntSwap {
  static inline void compareAndSwap(int* data) {
  if (data[I] > data[J])
    std::swap(data[I], data[J]);
  }
};

template<int I, int J>
class IntBubbleSortLoop {
 private:
  static const bool go = (J <= I-2);
 public:
  static inline void loop(int* data) {
    IntSwap<J,J+1>::compareAndSwap(data);
    IntBubbleSortLoop<go ? I : 0, go ? (J+1) : 0>::loop(data);
  }
};

template <>
struct IntBubbleSortLoop<0,0> {
  static inline void loop(int*) { }
};
```

```

template<int N>
struct IntBubbleSort {
  static inline void sort(int* data) {
    IntBubbleSortLoop<N-1,0>::loop(data);
    IntBubbleSort<N-1>::sort(data);
  }
};

template <>
struct IntBubbleSort<1> {
  static inline void sort(int* data) { }
};

int main() {
  int a[] = {3, 1, 2, 5};
  IntBubbleSort<4>::sort(a);
  for (int i = 0; i < 4; i++)
    std::cout << a[i] << " ";
  std::cout << std::endl;

  a[0] = 100;
  IntBubbleSort<4>::sort(a);
  for (int i = 0; i < 4; i++)
    std::cout << a[i] << " ";
  std::cout << std::endl;
}
```