Inline functions

Inline functions / methods

- A **hint** to a compiler to put function's code inline, rather than perform a regular function call.
When the compiler must produce an address of the function, it will always reject our request.

- Objective: improve **performance** of **small**, frequently used functions.

- An inline function defined in .cpp file is not recognized in other source files.

C vs C++ : macro vs inlining

compare:

```
define SQUARE(x) ((x)*(x))
SQUARE(i++) // unexpected behavior
```

to

```
inline int square(int x) { return x*x; }
square(i++) // good behavior
```

Tradeoffs:
Inline vs. Regular Functions / Methods

- Regular functions – when called, compiler stores return address of call, allocates memory for local variables, etc.

- Inline functions – no function call overhead, hence usually faster execution (especially!) as the compiler will be able to optimize *through* the call ("procedural integration").

- Inline functions - code is copied into program in place of call – can enlarge executable program

- Inline functions - can enlarge compile time. You compile the inline function again and again in every place it's used.