Interface Principles

Michael Henry
What is an Interface?

• An interface is the “middle man” between the code that provides a service and the code that uses it.

• An interface defines what some body of code does for its users.

• It also defines how member functions can be used by the rest of the program.
Four Golden Principles

• Hide Implementation Details
• Keep it Simple, Stupid
• Keep to Yourself
• Be Consistent
Hide Implementation Details

• An interface should hide details of the implementation that are irrelevant to the client (user) of the interface.
Keep it Simple, Stupid

• An interface should provide as much functionality as necessary, but no more.

• The functions of an interface should not overlap excessively in their capabilities.

• Do not provide multiple ways of doing the same thing.
Keep it Simple, Stupid

- The C standard IO library provides at least four different functions that will write a single character to an output stream:

  - char c;
  - putc(c, fp);
  - fputc(c, fp);
  - fprintf(fp, "%c", c);
  - fwrite(&c, sizeof(char), 1, fp);
Keep to Yourself

- A library function should not write secret files and variables or change global data, and it should be circumspect about modifying data in its caller.
Be Consistent

• Related things should be achieved by related means.

• Related member functions should behave similarly, accepting similar arguments and returning similar data types.

• Follow guidelines for programming style.
Use Consistent Naming

• Active names for functions, e.g. get_time
• Constants are written with capitals, e.g. INITIAL_SIZE
• Private variables start with an underscore, e.g. _size
Questions?