Try and Catch Blocks Solutions

Catch Blocks

- When is it useful to have more than one catch block after a try block?
  - When the code in the try block can throw exceptions which have different static types
  - Each catch block handles a different static type
- If more than one of the catch blocks is able to handle a certain exception, how does the program decide which one to invoke?
  - It will invoke the first catch block (in source code order) that matches the exception's static type

Multiple Catch Statement Example

- Consider the code sample on the following page
- If an exception is thrown, which catch block will handle it?
  - The exception is handled by the std::exception catch block
- Explain your answer
  - An exception is always caught by the first catch block which can handle it

Multiple Catch Statement Example

```
try {
    vector<int> v;
    cout << v.at(2) << endl;          // May throw an exception of type std::out_of_range
}
catch (const exception& e) {
    cout << "std::exception\n";
}
catch (const out_of_range& e) {
    cout << "std::out_of_range\n";
}
```

Writing an exception handler

- When writing an exception handler, how should the exception object be passed to it? Give a reason for your answer
  - The exception should be caught by reference (to const, unless the handler needs to modify it)
  - Catching by reference avoids making a copy of the exception object
  - It allows dynamic binding to be used when calling the exception object's virtual member functions

Writing an exception handler

- What guidelines should we follow when writing an exception handler?
  - Keep the code simple
  - Remember the program may not be in a stable state
  - Avoid creating new variables (especially if not built-in types)
  - Avoid allocating memory
  - Avoid calling functions
  - Avoid any code that might throw a fresh exception

Nested try/catch blocks

- Write a simple program which has a nested try/catch block
- Throw an exception which can only be handled by an outer catch block

Catching exceptions in a different function

- Write a simple program which calls a function
- The function throws an exception inside a try block
- The exception can only be handled by a catch block which is in the function's caller