

## **Goto: Start**

*"You guys are asking for  
Peanuts...and some carrots"*

*- Prof. Stephen Figlewski, Ph.D*


## **You Guys Know Us By Now...**

- We're Cool
- We're Smart
- We always rock the house
  
- DVD Today get it from Arvindh
  - You have to be a member. You can become a member by going to Arvindh

## Thanks DRP

---

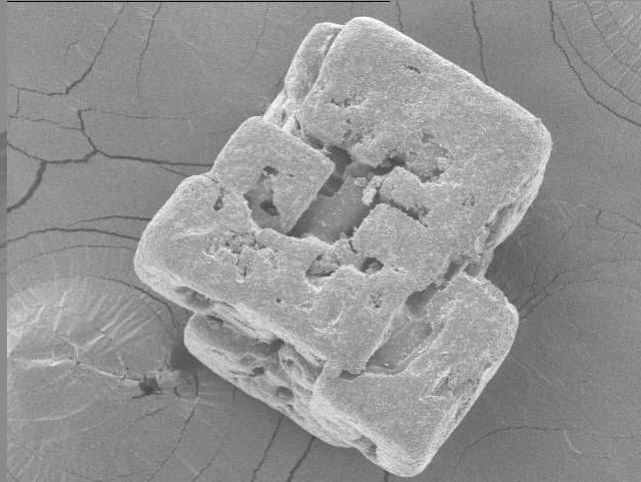
- Wow.
- Wow.
- Wow.
- Thanks DRP!
- Igor → Say some parting words
  
- And With That...Dr. Robert DeBellis



## *The 'Tech' Quant Interview*

or How I Learned to Stop Worrying and Love C++

Robert DeBellis, Ph.D.  
NYU QuantFS  
April 24, 2007



## A Fable

"Someone" was interviewing for a C++ development position at a hedge fund, and met with several department heads.

During one interview, the **Head of Development (CS MEng)**, commented on the interviewee's phone screen, take home exam, and in-house written exam. He claimed that the responses were: "...some of the best I've ever seen..."

During the next interview of the day, the **Head of Quant Research (Math PhD)**, commenting on (one supposes) the same evaluation methods claims: "You're C++ is...very poor".

*The Moral?*

## Things to Know

1. **an object oriented language (C++, Java)**
  2. a scientific language (Matlab or Mathematica)
  3. “statistical” software (R, Excel/VBA, SAS)
  4. data structures (STL is fine)
  5. database basics (SQL)
  6. software development
  7. design patterns
- Knowing some Linux doesn't hurt

## Stage 0 – The take-home test

- Only used occasionally (mostly by hedge funds).
- HARD
- Answers can be found on web.
- Make sure your answers are clear. Grammar counts.
- Topics: memory management, development/integration, migration (C vs C++), deep fundamentals

## Stage I – The Phone Interview

- Easy to Intermediate Questions
- The idea is to show prowess and demonstrate explanatory skills – can you “talk the talk”?
- Guaranteed Question: What sort of programming have you done?
- Hint: use a cheat sheet



## Phone interview continued...

- Vocabulary: lots of stuff...
- How does C++/Java implement object oriented design? How is polymorphism accomplished?
- What is a virtual function?
- Explain “is-a” versus “has-a”.
- Why make things const?
- Empty classes have what generated by default?
- What is an abstract class
- What’s the purpose of a namespace?
- What is a template? Why use one?
- Why use const variables versus macros?
- When is memory set aside?
- What is the difference between overloading and overriding?
- How would you model something?
- Difference between a reference and pointer? When to use either? What is a smart pointer?
- What does static mean?
- Data structure/STL trivia: how is a list different from a vector?
- Explain (insert data structure here)

## Stage II - The in-house written exam

HARD, but formulaic

1. "debug this code" - doesn't compile
2. "debug this code" - compiles but wrong output
3. "debug this code" - compiles but gives weird memory output (find the pointer error)
4. Here's some code that is supposed to do something. Find all of the errors (there are a lot of them)
5. Here's some code that works correctly. Write some functions/classes/templates to extend it to do something more
6. Write some code to accomplish something based on specifications.

## In-house exam continued...

"What does this code output?"

The usual suspects:

- recursion
- order of operations (esp. pre and post-fix)
- string manipulation (C and C++ styles)
- file i/o and streams
- arrays and pointers

## Stage III – In person

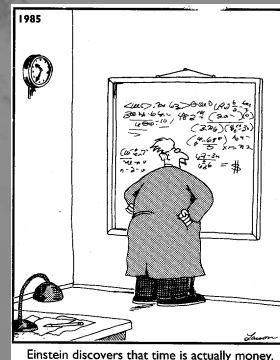
- Conversational
- Deep details of what you've worked on
- How would you go about solving such-and-such problem?
- Why would you choose one strategy over another?

Depth of knowledge probed

## Stage III – in person “blackboard” problems

- prototype a class/model
- overload pre and post-fix operators
- write a deep copy
- memory management
- algorithms
- design patterns

Sometimes a “psychological” test



Trite, but true

## Be prepared!

O'Reilly C++ Pocket Reference by Loudon  
<http://newdata.box.sk/bx/c/>: Teach Yourself C++ in  
21 Days:

<http://www.codeproject.com/>

<http://www.techinterviews.com/>

<http://the-name-less-blog.blogspot.com/>

<http://www.geekinterview.com/>

<http://www.wilmott.com/>

<http://home.earthlink.net/~huston2/dp/patterns.html>:  
Design Patterns

## The End

