# CS 216 Exam 1

You MUST write your name and e-mail ID on EACH page and bubble in your userid at the bottom of EACH page – including this page.

If you are still writing when "pens down" is called, your exam will be ripped up and not graded – even if you are still writing to fill in the bubble forms. So please do that first. Sorry to have to be strict on this.

Other than bubbling in your userid at the bottom, please do not write in the footer section of each page.

There are 10 pages to this exam – once the exam starts, please make sure you have all 10 pages.

There are three types of questions: short answer (worth 3 points each), medium answer (worth 6 points each), and long answer (worth 12 points each). The short answer questions should not take more than a line or two to answer – *your answer should not exceed about 20 words.* There are 99 points of questions and 1 hour 35 minutes (95 minutes) to take the exam, which means you should spend just under one minute per question point. This test is worth 100 points – the remaining 1 point you get for filling out your name, e-mail id, and bubble sheets on the last page.

**If you do not bubble in a page, you will not receive credit for that page! If the page is worth zero (or one) points, then you will receive a grade penalty.**

This exam is CLOSED text book, closed-notes, **closed-calculator**, closed-cell phone, closed-computer, closed-neighbor, etc. Questions are worth different amounts, so be sure to look over all the questions and plan your time accordingly. Please sign the honor pledge here:

_____

_____

_____

_____

*There are 10 types of people in the world – those that understand binary and those that don't.*

---

(the bubble footer is automatically inserted in this space)

## C/C++

1. [3 points] What are the three properties of references that distinguish them from pointers? Declaration syntax is not what we are looking for here – you need to come up with 3 other differences.

2. [3 points] When manipulating an object, when is the copy constructor called versus the `operator=()` method?

3. [3 points] What is the purpose of a destructor method? What does it do?

4. [3 points] Array base names are just like pointers, expect in one respect that we have seen. What is it?

---

5. [3 points] When would we *not* use the `explicit` keyword in C++? Answering "when we want implicit …" is not what we are looking for here – describe a scenario where we would specifically not want to use the `explicit` keyword.

6. [3 points] When would we want to pass a pointer by reference?

7. [6 points] When we are creating header files (i.e., List.h), we often put them in a `#ifndef/#define/#endif` block. What is the syntax for that block (write out a sample set of `#ifndef/#define/#endif`)? Why do we do this? Lastly, how does it work?

---

8. [12 points] Consider the declaration for a 2-D array in C++: `int foo[4][3];` We'll assume that it is properly initialized (one way or the other) to have 12 values. Diagram what memory looks like for this array. Where would `foo[2][2]` be (meaning what memory address would it be found at, relative to where `foo` is)?

## Lists

9. [3 points] What do templates do in C++? Why are they needed?

10. [3 points] What does the `friend` keyword do in C++?

11. [3 points] Other than a postfix calculator, name 2 applications of a stack.

12. [3 points] What is an abstract data type (ADT)? What is contained in an ADT? When would we use an ADT?

---

## Numbers

13. [12 points] Consider a few primitive data types in C++. We will define the `byte`, which is a 8 bit 2's complement integer (just like the 2's complement numbers we've seen in class, but with 8 bits instead of 32 bits). We will also define the `oneint` type, which is an 8-bit integer stored in *1's complement* form. Now consider the following code:

```
union {
   byte b;
   oneint i;
} foo;
foo.b = -27;
cout << foo.i << endl;
```

What get printed? Show your work!

---

14. [3 points] What is fixed point number representation? List one pro and one con for that format.

15. [3 points] Floating point numbers have issues with numerical precision, as discussed in lecture. List 3 ways that one can increase the precision of floating point numbers.

16. [3 points] Floating point numbers are not spatially uniform. What does this mean?

17. [3 points] The IBM floating point specification is similar to the IEEE 754 architecture that we studied in class, but instead of 8 bits for the exponent, it uses 7. Similarly, instead of 23 bits for the mantissa, it uses 24. What are the advantages to the IBM floating point specification over IEEE 754? What are the advantages of IEEE 754 over IBM floating point? List one advantage of each.

---

18. [12 points] Convert the number 5.5 to IBM floating point notation (the notation is described in the previous question on the previous page). Your answer should be left in *little-endian* hexadecimal format. You should pick the exponent offset – if you don't know what it should be, take your best guess; either way, clearly specify what your exponent offset is, as there will be partial credit for this question.

---

## Big-Oh

19. [3 points] What does big-theta mean? Give two explanations for this one: one formulaically, and one as an English explanation.

20. [3 points] Why do we like big-theta better than big-oh?

21. [3 points] Why is there no little-theta?

22. [3 points] If $f \in O(g)$ and $g \in \Omega(h)$, what can we say about the relation between $f$ and $h$?

23. [3 points] Why don't we care about the base of logs (i.e. base 2 or base 10) when describing a logarithmic running time algorithm?

---

This page unintentionally left blank.

However, you get one free point for bubbling in this page.

---

(the bubble footer is automatically inserted in this space)