

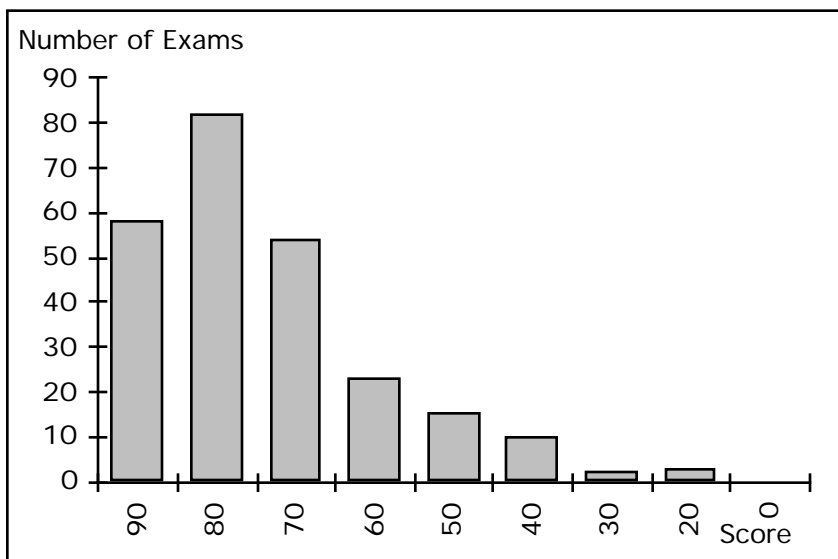
CS193i Final Exam + Solution

Exam

The exam is 2 hours long and is open book, open note, etc. Your exam should have 14 pages. Strategically, try not to get bogged down on a single problem— move on and come back to it if you have time. No problem should take more than 20 or 30 minutes. Good luck and have a nice summer!

Solution

This is the Spring 96-97 CS193i final exam with solutions and notes on the grading added to the text of the exam itself. The distribution of the scores is below -- the exam was moderately difficult: there were 23 exams with scores of 95 or above, although there was an enormous range in the scores. There was one person who got a 100.



median	82
mean	78.60
standard dev	15.02

1) TCP/IP Coding (20 points)

As background, here is the interface to the standard socket `read()` as it appears on our socket handout...

```
int read(int sock, char* buff, int buffLen);
```

Read bytes from a socket into the given buffer. Returns the number of bytes read, or returns -1 and sets `errno` on error. A return of 0 is an effective EOF— there are no more bytes. `read()` will not necessarily fill the entire buffer every time it is called. Typically, many successive calls to `read()` will be required to get all the bytes, either because they will not all fit in the given buffer, or because not all the bytes have arrived the first time `read()` is called.

One problem with the standard socket `read()` is that it is not convenient for reading lines of text. Ordinarily, each call to `read()` returns some unpredictable numbers of characters — possibly a fraction of line, or possibly several lines lumped together. For this problem you will write a `read_line()` function specialized to read text data one line at a time from a socket.

`read_line()` is like `read()`, except it tries to return a buffer containing exactly one line of characters up to and including a `'\n'` from the socket. Like `read()`, `read_line()` returns the number of bytes read, or 0 if there are no more bytes, or -1 if there is a socket error. As an additional error case, `read_line()` returns -2, if `buffLen` characters have been read into the buffer, but without yet seeing a `'\n'`. Like `read()`, `read_line()` should not null terminate the buffer. The simplest implementation strategy is to call `read()` repeatedly to retrieve 1 character at a time.

```
int read_line(int sock, char* buff, int buffLen) {
```

Answer

There were many ways to get a fully functional `read_line` but here's one correct example:

```
int read_line(int sock, char* buff, int buffLen) {
    int i, result;
    char ch; //so you don't need memory allocation

    for(i=0; i<buffLen; i++)
    {
        result = read(sock, &ch, 1); //&ch to get a char*

        if(result<=0) return result;
        buff[i]=ch;
        if(ch=='\n') return i+1;      //i+1 to account for the '\n'
    }
    return -2;
}
```

Some people tried to read a full `buffLen` and then find a `'\n'` within the buffer. However, this approach doesn't work for the subsequent calls to `read_line` without unpleasant global state saving. Such "over-reading" of the socket was penalized with a -10.

Our grading criteria was divided into 2 parts : 8 points for return handling and 12 points for read loop correctness. Some common problems with the read loop involved several potential misunderstandings of the behavior of `read`. Such problems included passing a `char` to `read` or a `char *` with no allocated memory (-2). A lot of people also assumed that `read` returned a `'\0'` terminated string (required for functions such as `strcat`) (-4/-6). Finally

there were also several occurrences read_line that attempted to manipulate the characters in a local buffer of size buffLen and then simply set buff to point to that local buffer not realizing that only the local copy of buff is affected(-4).

There were 4 return cases to handle (encountered '\n', buffer full, no more character to read, socket error) so each was assigned 2 points where one had to both detect and return correctly in order to get the credit. In addition, any major C error was assigned a small penalty (-1/-2) and minor C errors were just ignored). For the case where read returns 0 (EOF) before read_line gets a '\n', either 0 or -2 were allowable return values.

2) Miscellaneous **Short Answer**. 1 word or 1 sentence answers— don't bog down with partial-credit explanations. If you know it great, otherwise leave it blank. There are 16 sub-parts. (25 points)

These were graded -1 for small errors, -2 for larger errors.

a) What is the difference between GET and HEAD in HTTP?

GET returns the HTTP header followed by the document. HEAD only returns the HTTP header.

b) Each of the following URL's is href'd in the page <http://hell.org/doc/punish.html>

For each URL, give its absolute form to the right...

- i) `/binky.html`
- ii) `ironic/homer.html#doh`
- iii) `donut.gif`
- iv) `/test.html#contents`

Key points: understand how relative URLs work, realize that #doh is part of the URL.

i) `http://hell.org/binky.html`

ii) `http://hell.org/doc/ironic/homer.html#doh`

iii) `http://hell.org/doc/donut.gif`

iv) `http://hell.org/test.html#contents`

Cap of -5 for the whole thing.

c) What is the complete, exact string which the HTTP client sends to the HTTP server to request page (iv) above? Give your answer as a double quoted (") string constant as you would in C or Perl...

`"GET /test.html HTTP/1.0\r\n\r\n"`

Key points: remember how to put together an HTTP request which SiteSucker covered rather extensively.

d) How does an HTTP client know the type of the data returned by the server?

The Content-Type: field in the HTTP header returned by the server.

e) X is the superclass of Y. Is the assignment below a compile time error?

```
X x = new X;
Y y = new Y;

y = x;    // Is this a compile-time error?
```

Yes, this is a compile time error. Subclasses have the properties of their superclasses and can therefore stand in for their superclasses. Not the reverse.

f) X is the superclass of Y. Both classes define a void foo() method. On the "x.foo()" message send -- which method definition is invoked: the X class' or the Y class'?

```
X x = new Y;

x.foo(); // which method does this invoke?
```

Invokes the y.foo() method. This is the "objects always remember their class" rule in action.

g) Once again, X is the superclass of Y. What is the output of the Y.demo() method?

```
class X {
    void test() {
        System.out.println("X test");
        foo();
        bar();
    }

    void foo() {
        System.out.println("X foo");
    }

    void bar() {
        System.out.println("X bar");
    }
}

class Y extends X {
    void test() {
        System.out.println("Y test");
        super.test()
    }

    void foo() {
        System.out.println("Y foo");
    }

    static demo() {
        Y y = new Y;
        y.test();
    }
}
```

Y Test

X Test

Y Foo (this is the most important line)

X Bar

h) Microsoft has the MFC C++ class library for building Windows GUI applications. Metrowerks has the PowerPlant C++ class library for building MacOS applications. Both of these are more capable in many ways than Java's AWT. What are two key advantages of AWT compared to these traditional programming tools? (Or put another way: why would anyone want to use AWT?)

Many people wrote just one reason, or wrote two reasons which were really one reason expressed twice (-1). For full credit, it was important to mention things which are true about Java/AWT but which were not true about C++ and its librarians. Three possible reasons are...

a) *Portable. Java+AWT have the potential to work without recompilation across many platforms.*

b) *Programmer Efficiency. Java includes programmer efficiency features (GC, RTTI, archiving) which are better developed than in C++.*

c) *Web-Enabled. Java/Awt include features (security, the Applet class) which help them operate in a client/server environment.*

i) Most of our current Internet technology is based around public standards which have grown to have high quality implementations. Historically, many proprietary standards have not been widely adopted or well implemented. Why do public standards tend to eventually have better implementations?

Competition among multiple implementations. The marketplace of multiple implementors drives the quality of the implementations available up. There's a related virtuous cycle — because people expect the public standard to be better supported, they support it.

j) When you give your browser a URL such as "http://www-cse.stanford.edu/" — most likely what host must the browser first contact as part of retrieving the page (somewhat tricky).

The local DNS server to look up the IP addr of www-cse.stanford.edu. One person cleverly pointed out that technically, the most likely computer contacted will be the first router on the path towards the local DNS server, although we could dispute whether the router counts as a "host".

k) What are the two main things encoded in an IP address?

The host # and its network #.

l) Bob Metcalf and Larry Wall are two very influential people for the content that has become CS193i. Which one of them invented Ethernet?

Bob Metcalfe. A little additional research suggests that there's an 'e' at the end of Metcalfe, although both spellings are used widely on the net!

m) What prevents IP datagrams from getting stuck in "infinite loops" in the Internet — forwarded from X, to Y, to X, to Y...

The Time-To-Live counter in each datagram. "Routers" or "TCP/IP" were not sufficiently specific.

n) True or False. When an IP router receives an IP datagram, it looks at the IP address of the final destination and figures out all the routers the datagram will need to traverse on its way.

False. It is largely concerned just with the "next hop".

o) Here is some HTML source.

```
<p>This dress <p>exacerbates the genetic
betrayal
that is <p> my legacy
```

Please draw how the HTML might render in a browser window which is..

| <----- this wide -----> |

This dress
 exacerbates the genetic
 betrayal that is
 my legacy

Key points: realize that whitespace in the HTML source is ignored. Wrap the HTML appearance to the browser width, and put in vertical whitespace (wt) between the three paragraphs. The quote is from the embittered Janeane Garafolo in "Romy and Michele's High School Reunion "

p) Currently, you can be more confident that a Java applet is not going to erase your hard drive than an ActiveX control -- name one technical difference that makes Java applets safer.

Features which are true about Java which are not true about ActiveX: Java has a form of interpreter which guards run-time actions, Java Applets do not have arbitrary system call or file system access.

3) CGI Coding (25 points)

For this problem, you will write an AppendDB variant of WebDB which does input, but has no display capability. As on the homework, we will provide the standard CGI structure, and you need to fill in the correct code in a few places.

AppendDB should support adding a record to a tab delimited text file named in the global variable \$fname. The database should have the field names on the first line, and all the records on the subsequent lines. When invoked with no input, AppendDB should respond with a form featuring a <textarea> for each field. In the form, each field text area should be immediately preceded by its name and followed by an <hr>. AppendDB should not have the number or names of the fields hardcoded — it should retrieve them by reading the first line of the data file. When the form is submitted, AppendDB should add the record to the data file using append mode and return a "Record Added" message to the client.

Below, we provide most of the standard structure for AppendDB — you need to fill in the correct Perl code in a few places...

```

/usr/bin/perl
require "193icgi.pl";

$fname = "movies.db";    # this could name any tab-delimited text db

$header = <<END_TEXT;
<html><head>
<title>AppendDB</title>
</head>
<body>
END_TEXT

$strailer = "</body></html>\n";

#Begin our HTTP response...
#####
print "Content-type: text/html\r\n\r\n";
print $header;

%pairs = &ExtractPairs;

```

##a) At this point, in all cases AppendDB needs to know the names of the fields. Write the code to open the data file, read the tab separated field names from the first line, build a global @fields array which contains the field names, and close the data file.

```

open (DATAFILE, "$fname");
chop ($line = <DATAFILE>);
@fields = split (/\\t/, $line);
close (DATAFILE);

```

Common mistakes:

- forgetting to chop off the trailing \n (no pts. taken off)
- using index (instead of split) to parse out the fields
 - forgetting that the last field is not followed by a \t and won't be found with index
- reading in all the lines -- only the first line is needed, since only the first line contains the field names

Grading: (5 pts.)

1 open, 1 reading in first line correctly, 3 parsing

```

if (scalar(%pairs) == 0) {
  ##b) In this case, AppendDB should respond with a form including a <textarea> for
  each field in @fields.

  print "<form>\n"
  foreach $field (@fields) {
    print "$field\n";
    print "<textarea name=$field><\textarea>\n";
    print "<hr>\n";
  }
  print "<input type=submit\n";
  print "<\form>"

  print $trailer;
  exit(0);
}

```

Common mistakes:

- a) using <input type=text> instead of <textarea>
- b) not using a distinct name for the fields so you can pick them out later (either \$field, or \$count, where \$count keeps track of the number of fields) for each textarea
- c) forgetting to print the field name or <hr>

Grading: (8 pts.)

- 1 printing <form> and <\form> correctly
- 2 for-loop to iterate over all the fields
- 1 correct format: fieldname, textarea, <hr>
- 3 <textarea ...> correct: used textarea, remembered <\textarea>, had something valid for the name
- 1 submit

```

else {
  ##c) In this case, AppendDB should append the text entered in the form fields as
  a new tab-delimited record in the database and print a confirmation. Use append
  mode to open the file.

  open (DATAFILE, ">>$fname");
  $line = "";
  foreach $field (@fields) {
    $line .= "$pairs{$field}\t";
  }
  chop ($line);
  print DATAFILE "$line\n";
  close (DATAFILE);
  print "Record Added\n";

  print $trailer;
  exit(0);
}

```

Another good solution was to push each value associated with a field name onto an array and then joining with \t.

Common mistakes:

- a) iterating through all the keys in pairs (this will include "submit" and does not guarantee order of keys will be order of field names) instead of iterating through all the field names
- b) forgetting to chop off the last \t
- c) not using the associative array \$pairs correctly

Grading:

- 2 opening with append
- 4 for-loop through field names (or index over field names)
- 2 getting the value for a given field through \$pairs
- 3 output formatting:
 - 1 trailing tab
 - 1 omitting ending newline
- 1 "Record added" message

4) Java Coding I (20 points)

For this problem, you will define two of the world's more useless Java Canvases. Your job is to define two types of Canvas: XHello and XRect. The two classes are similar in most ways...

Both have a color instance variable which initially is Color.black

Both respond to paint(Graphics g) as is usual for Components and Canvases. For their appearance, both classes should present a big "X" made of two lines in the current color to fill their current size (g.drawLine(x1, y1, x2, y2;). We will use the simplifying convention that Components do not need to restore the foreground color after setting it, so you may g.setColor(...) in paint() to whatever you like, and not worry about restoring it afterwards. Any Component can retrieve its current size by sending itself the size() message...

```
Dimension dim = size();    // sets dim.width and dim.height
```

Both respond to setColor(Color newColor) to change their current color

For simplicity, there are no arguments for the constructor. (In fact, you can get away without a constructor at all, just so long as the initial color is Color.black) The client should send resize() and setColor() after the object is built to change its state.

The two classes are different in that...

XHello: In addition to the big "X", XHello draws the string "Hello" at (20,20) (g.drawString("Hello", 20, 20;). Just like the X, the string should appear in the current color.

XRect: In addition to the big "X", XRect draws a rect at (0, 0) with width and height of 20 (g.drawRect(0, 0, 20, 20;). The rect should be in the current color.

On the client side, we might have code which looks like...

```
public class TestApplet extends Applet {
    public void init() {

        XHello x1 = new XHello();
        x1.resize(100,100);
        add(x1);

        XRect x2 = new XRect();
        x2.resize(40,40);
        add(x2);
        x2.setColor(Color.red);

    }
}
```

Define the XHello and XRect classes. You should create a common superclass X between the two to help factor out all their common behavior. You may define X as being abstract or not. Your goal is to eliminate any code duplication between XHello and XRect. To save time, you may omit the public and. private specifiers, you may assume that all the imports are done for you, and you may ignore what classes are in what files.

Define the X class on this page as a subclass of Canvas.

```
public class X extends Canvas
{
    private Color mColor = Color.black;

    public void paint(Graphics g)
    {
        g.setColor(mColor);
        Dimension dim = size();
        g.drawLine(0, 0, dim.width-1, dim.height-1);
        g.drawLine(0, dim.height-1, dim.width-1, 0);
    }

    public void setColor(Color color)
    {
        mColor = color;
        repaint();
    }
}
```

Issues

- implement paint()
- call repaint() in setColor() to keep appearance and data model in synch

Define the XHello and XRect classes on this page as subclasses of X.

```
public class XHello extends X
{
    public void paint(Graphics g)
    {
        super.paint();
        g.drawString("Hello", 20, 20);
    }
}
```

```
public class XRect extends X
{
    public void paint(Graphics g)
    {
        super.paint();
        g.drawRect(0, 0, 20, 20);
    }
}
```

Issues

-Take advantage of code sharing between X, XHello, and XRect. Either call paint() in superclass, or alternately keep the common behavior factored into an X method such as PaintUtil().

5) Java Coding II (10 Points)

For this problem, you will code a little threaded client for the X classes. You do not need to have completed the X problem to get credit for this problem — for this problem we assume that there is a correct X implementation available.

a) Write a Morph subclass of Thread with the following features...(space on the next page)

Morph keeps an array of references of up to 10 X objects. Morph should also keep track of how many X's it has.

Morph responds to an add(X x) message which adds an X to its array. The order of the objects in the array is not significant.

In its run(), Morph has the following behavior: go through all the X's, and setColor() them to Color.red. Then go through all the X's and setColor() them to Color.black. Continue alternating the colors indefinitely.

After every setColor() it sends, Morph should try to sleep for 50 milliseconds with the standard phrase...

```
try{ sleep(50); }  
catch(Exception e) {}
```

You do not need to worry about synchronization for Morph — it turns out that, as defined, Morph does not have very significant synchronization issues anyway.

(space for your Morph definition code)

```
public class Morph extends Thread
{
    private X mXArray[] = new X[10];
    private int mNumX = 0;

    public void add(X x)
    {
        if(mNumX < 10) mXArray[mNumX++] = x;
    }

    public void run()
    {
        Color currColor = Color.red;
        while(true)
        {
            for(int i = 0; i < mNumX; i++)
            {
                mXArray[i].setColor(currColor);
                try{ sleep(50); }
                catch(Exception e) {}
            }
            currColor = (currColor == Color.red) ?
                Color.black :
                Color.red;
        }
    }
}
```

Issues

- implement an add() which correctly adds X's to the array
- implement a run() which goes through and messages all the X's

b) Below is the code from Java Coding I to create a couple X objects in an Applet. Extend the code using a single Morph object to alternate the colors of the two X objects..

```
public class TestApplet extends Applet {
    public void init() {

        XHello x1 = new XHello();
        x1.resize(100,100);
        add(x1);

        XRect x2 = new XRect();
        x2.resize(40,40);
        add(x2);
        x2.setColor(Color.red);

        // Add code below to introduce a Morph object on x1 and x2

        Morph morph = new Morph();
        morph.add(x1);
        morph.add(x2);
        morph.start();
    }
}
```

Issues

- Create a new Morph thread object
- Message it to add two X's
- Call morph.start() (not morph.run())