University of Arizona, Department of Computer Science

CSc 520 — Final Exam — 50%

Christian Collberg
April 14, 2008

1 Introduction

This is your final exam for 520. The idea is to pick a language, research it, write up your results, and present to the class. You will be graded as follows:

1. A set of slides to be used during the presentation (10%, due noon, April 30),
2. A small program written by yourselves (10%, due noon, April 30),
3. A 10 minute presentation (10%),
4. A 10-page report (20%, draft due noon April 30, final version due 4pm, May 14).

This project should be done in groups of two students.

The final page of this handout has a form for changing the assessment scheme of the class. If you agree with it, please fill it out and hand it in to me.

2 The Languages

In Table 1 I give the languages we're going to study. I've assigned students randomly to languages and grouped students based on who they've worked with on the other assignments. You are allowed to swap partners and languages if you wish — just let me know. If you have a special interest in a language not on the list you can try to persuade me to let you work on that one.[1]

3 The Program [10%]

You should write a program that shows off the important features of the language. Due date is April 30. The program doesn't have to be long, maybe a page or two of code depending on the language, but it should illustrate what the language is good at.

Your program will be graded on

1. correctness,
2. how interesting it is,
3. how well it illustrates the basic ideas of the language.

[1] If you already know this language, or if it's C++, I'm unlikely to let you work on it!

Table 1: Project languages.

#	Language	URL	Presentation date	Group
	Object-Oriented Languages			
1	Eiffel	`www2.eiffel.com/download`	April 30	anurag jieyao
2	Objective-C	`www.gnu.org/software/gcc`	April 30	karthikr bhavinbm
3	Modula-3	`www.modula3.org`	April 30	bojan swami
4	Oberon	`www.oberon.ethz.ch`	April 30	bsampath manish
5	C#	`www.mono-project.com/Main_Page`	April 30	anandp deeptis
	Stack languages			
6	FORTH	`www.forth.com/forth/index.html`	April 30	russelll harleycw
7	Postscript		April 30	sushanth rsheshu
	Miscellaneous languages			
8	LOGO	`www.cs.berkeley.edu/~bh/logo.html`	May 5	ricarlos jsamdal
9	SETL	`cs.nyu.edu/~bacon/download-setl.html`	May 5	tpatki bhandari
10	FORTRAN 95	`gcc.gnu.org/fortran`	May 5	xingqu esgreene
	Functional Languages			
11	O'caml	`caml.inria.fr`	May 5	qtung pavank
12	APL/J	`www.jsoftware.com`	May 5	qingju kims
13	Erlang	`www.erlang.org`	May 5	rbailey iryan
	Scripting Languages			
14	Python	`www.python.org`	May 7	rtorres fcirett
15	Perl	`www.perl.org`	May 7	marshall ghigliom
16	Icon	`www.cs.arizona.edu/icon`	May 7	parags lopa
17	gawk	`www.gnu.org/software/gawk`	May 7	amit nithya
18	JavaScript	`www.njs-javascript.org`	May 7	natashag pallavic
19	TCL/TK	`www.tcl.tk`	May 7	balajir xuchenw

4 The Slides [10%]

You should have around 10 slides (no less than 8, no more than 12) which you must submit (in PDF) by April 30. You're encouraged to come to my office and show what you've got before your talk.

You can use the program you wrote as illustration in the slides, or, maybe better yet, write several small program fragments as illustration.

The first slide (not counted among the 10) should be a title slide, containing the name of the language and the name of the students in the group.

Note that your professor is not impressed by fancy layouts, green text on black background, useless animations, > 17 words per slide — make your slides simple, clean, elegant, and informative! Use large black text on a plain white background. Don't overload the slides with information.

You will be graded on

1. layout, choice of fonts, choice of background, general prettiness, etc.
2. organization,
3. choice of information.

5 The Presentation [10%]

The presentation should take 10 minutes, not more, not less. I will use a timer and cut you off — in mid-sentence, if necessary — if you'run over. You must therefore rehearse several times by yourselves to make sure you'll stay within the limits.

It's a good idea to pair up with another group to practice by giving the presentations to each other.

The presentation should cover the following:

1. History of the language (who designed it, when, what other languages it is related to, etc.).
2. What is the target audience of the language (who should write in it, what kinds of programs is it meant for, etc.)?
3. What are the most interesting aspects of the language (what makes it different from other languages)?
4. Show a small program you've written.
5. Pick a few of the concepts (typing, iterators, memory management, orthogonality. . .) we've discussed in class and describe how they relate to the language.

Don't talk about things that the audience either knows, or would not be interested in. Saying "an if-statement looks like this, a procedure call uses these kinds of brackets" is likely to put your audience to sleep. Use terminology and ideas we've discussed in class.

You will be graded on

1. the fluency of your presentation,

2. the pace of the talk (not too fast, not too slow, not too much information, finish within the time allotted),
3. how well you've chosen interesting aspects to talk about,
4. how well you use terminology and concepts taught in class,
5. how well you appear to understand the language.

Don't cram as much information as possible into the talk — that's not the purpose. Instead, choose wisely what to present and what to leave out.

6 The Report [20%]

The report should be around 10 pages long. Use 11 points, 1 inch margins. A draft is due April 30, the final report is due May 14 (the day of the final exam). It should cover the same as the presentation, but in more detail. In particular, think of all the concepts we've discussed in class and how your language relates to them. Here are some of the concepts we've discussed:

1. Compilation vs. interpretation?
2. Value vs. reference semantics?
3. Iterators?
4. Scope rules?
5. Exceptions?
6. ...

Also study, if possible, the compiler/interpreter itself. Does it support garbage collection? Which algorithm does it use?

The report should be submitted in PDF. Don't forget the title (the name of the language) and the authors' names.

You will be graded on

1. correctness,
2. quality of presentation (grammar, structure, spelling),
3. how well you've researched the language,
4. how well you related the language constructs to the concepts taught in class.

7 So, what should I do?!?!?

Start by finding a compiler/interpreter for the language and install it on your machine. Look first at the links in Table 1, but you may find better implementations for your particular machine elsewhere, of course. Some of the languages have compilers installed on lectura already. This is true of Icon (`icont`), gawk, Objective-C (`gcc -x objective-C`), FORTRAN (`gcc -x f95`), perl, python, TCL (`tclsh`), and possibly others.

Next, find resources to read! Most of these languages will have manuals online that you can study, and some will have books you can check out from the library.[2]

Go through the documentation and, at the same time, write little mini-programs in the language to help you learn it.

Look on-line for "real" programs written in the language. Study them to learn more.

Make a list of the concepts we've learned in class. How does your language deal with each one of them?

Write your own little program.

Decide how you're going to do the presentation. Since you only have 10 minutes you have to organize it very carefully. Which concepts are you going to present? How much time on each of them? What's most special about this language? What would the audience most like to learn? If the language is similar to another one that you think the audience might know you can say "the type system is just like language X, except for. . . " and thereby avoid much boring detail. Show code snippets to illustrate your points, but only if you think the audience can understand them in a really short time. Drawings are good, better than words.

You may want to talk about the compiler/interpreter, too: is this language normally compiled or interpreted? Compiled to bytecode or interpreted directly? Garbage collected or not?

Make the slides. Simple is good. Readable is good. Make a PDF file out of the slides and submit. I'm going to preload my computer with every group's PDF slides so that we can quickly move from one presentation to the next. DO NOT use any features of Powerpoint (or whatever system you use to generate slides) that don't show up well in PDF. I will NOT let you load Powerpoint directly — we don't have enough time in between presentations.

Make a script. Since you only have 10 minutes to talk you should decide exactly what to say for each slide.

Decide who will say what. Practice. Practice some more. Time yourselves. Rewrite the script to fit the time. Practice again. Ask another group to listen to your talk and critique it.

Now, you should have no problem writing the draft paper. Try to get all the data in there (maybe as bullet points), but you can wait with the prose. (If you know LaTeX I suggest you use it rather than Word — it makes for much easier writing and prettier output.) Submit.

Finally, fix up the project document. It should be grammatical and clear. Easy to read. Informative. Use example code snippets to illustrate your points. If your native language isn't English (or, if it is, and you just can't write), find someone who can check your grammar for you.[3]

8 Submission and Assessment

The final exam is worth 50% of your final grade.

You should submit the assignment electronically using the `Unix` command

```
turnin cs520.final language-slides.pdf language-paper.pdf language-program.txt README
```
.

[2] A "library" is kind of like google or wikipedia, except you get so called "physical copies" of the texts without having to print them out yourself. It's quite a neat idea but I doubt it will ever catch on.

[3] Hire an English major. They're cheap.

Regarding CSC520, Principles of Programming Languages, Spring 2008

Change of assessment

I'm OK with changing the assessment of the course from having a two hour final exam (worth 50% of the course grade), to a final project consisting of

1. a 10-page research paper describing a programming language (20%),
2. a set of 10 presentation slides (10%),
3. a 10 minute presentation (10%),
4. a small program written in the language under study (10%).

name (sign): ______________________________

name (print): ______________________________

CS login (print): ____________

date: ________________