PLEASE HAND IN

UNIVERSITY OF TORONTO
Faculty of Arts and Science

St. George Campus

APRIL-MAY 2008 EXAMINATIONS

CSC 207H1S
Duration — 2 hours

PLEASE HAND IN

Examination Aids: **Any handwritten or printed materials. No electronic aids.**

Student Number: |_|_|_|_|_|_|_|_|_|

Family Name(s): ______________________

Given Name(s): ______________________

Do ***not*** *turn this page until you have received the signal to start.*
(In the meantime, please fill out the identification section above, and read the instructions below *carefully*.)

This final examination consists of 6 questions on 14 pages (including this one). *When you receive the signal to start, please make sure that your copy of the examination is complete.*
There is one mostly-blank page at the end of the exam that you may use if you run out of space.
All program code must be in Python. Comments are not necessary unless specifically indicated in the question.

MARKING GUIDE

# 1:	______/10
# 2:	______/10
# 3:	______/ 5
# 4:	______/10
# 5:	______/ 5
# 6:	______/10
TOTAL:	______/50

Good Luck!

Question 1. [10 MARKS]

Here are several executions of a program "geto.py", which runs from the command line, getting all the information it needs from the options and arguments. The commands typed by the user are shown in italics. The shell prompt is a single dollar sign, '$'.

```
$ geto.py -x
bad option
$ geto.py
Exactly one option must be set.
$ geto.py -s -p
Exactly one option must be set.
$ geto.py -s -n 2 -p 3 4
Exactly one option must be set.
$ geto.py -s 2 3
Sum of args: 5.0
$ geto.py -s 2
Sum of args: 2.0
$ geto.py -s 2 3 4 5.1
Sum of args: 14.1
$ geto.py -p 2 3
Product of args: 6.0
$ geto.py -p 2
Product of args: 2.0
$ geto.py -p 2 3 4
Product of args: 24.0
$ geto.py -n 3 2 4
Sum of 3.0th powers of args: 72.0
$ geto.py -n 0.5 2
Sum of 0.5th powers of args: 1.41421356237
```

Write a program using the **getopt** module to produce the output shown.

You may assume that the arguments provided by the user will always be correctly formatted **float** numbers where required. You may also assume that all the output shown is from the standard output, not the standard error stream.

This page gives you more space for your answer.

Question 2. [10 MARKS]

This question requires you to write a makefile to manage updating for a collection of six files. Here is how the files are related.

- `P.java` is a Java source file. It is only changed by human users.
- `P.class` is a Java class file produced by compiling `P.java`. The Java compiler (probably but not certainly "`javac`") is identified by the environment variable `JAVAC`, and the execution command (probably "`java`") is in the environment variable `JAVA`.
- `classes` is produced by executing the command

```
grep class P.java > classes
```

- `out1` is produced by executing `P.class` with the file `classes` as standard input and `out1` as standard output.
- `dataA` is produced and edited by human users.
- `out2` is produced by executing `P.class` with `dataA` as the first command-line argument and `out2` as the second command-line argument. (The standard input and standard output are not used.)

Part (a) [2 MARKS]

Draw a diagram showing the dependencies among the files in this question.

Part (b) [8 MARKS]

Write a makefile implementing the relationships among the files in this collection.

This page gives you more space for your answer.

Question 3. [5 MARKS]

Write an XML file that expresses the content of the rules stated in the previous question. Make up your own tag names.

You will be marked on the consistency and completeness with which you represent the information.

Do not worry about the difficulty of representing '<' and '>' in the XML context where those characters are ordinarily special. Use the same notation for environment variables as `make` does.

This page gives you more space for your answer, though you almost certainly do not need it.

Question 4. [10 MARKS]

A moderately unusual feature of this course has been the distribution of term work, with fewer large assignments and many more short exercises than in most Computer Science courses. State an opinion of this approach to term work, giving arguments for and against in separate paragraphs, and concluding with a summation that states an overall balance.

Your mark will depend on the quality and presentation of your arguments, not on whether the marker agrees with you or on whether they are truly your own opinions. (Nevertheless, the marker's own view of the term work will partly be formed by what you say; please be truthful unless you really think it will cost you marks.)

This page gives you more space for your answer. Do not feel obliged to fill it.

Question 5. [5 MARKS]

The Unix "find" command allows operations on specified files in an entire subtree of the file system. Here we just use it to list all the files in the directory "mydir":

```
$ find mydir -print
mydir
mydir/a.py
mydir/a.very.odd.filename.txt.data
mydir/alphabet.py
mydir/b.txt
mydir/c.out
mydir/subdir
mydir/subdir/e.txt
mydir/subdir/f.py
mydir/subdir/g.data
```

Your task is to write a program that takes its standard input from the standard output of a "find" command such as the one shown above and prints information for selected files. The selected files are those with a given extension (where the "extension" is the part of the file name after the last period, if there is one); the extension is specified as the first and only command-line argument. The information to be printed is the name of the directory containing the file followed by the name of the file itself.

For example, for the directory contents shown above, here is a possible set of interactions:

```
$ find mydir -print | python regex.py py
mydir a.py
mydir alphabet.py
subdir f.py
$ find mydir -print | python regex.py txt
mydir b.txt
subdir e.txt
$ find mydir -print | python regex.py data
mydir a.very.odd.filename.txt.data
subdir g.data
```

Your program must use a regular expression to extract the directory and file name from each line. Exactly one command-line argument is required, and you may assume that it is correctly given by the user.

Hint: Your answer will probably be quite short.

This page gives you more space for your answer.

Question 6. [10 MARKS]

Compilers for the C language incorporate a "preprocessor" that adjusts the text of the program in various ways before the program is compiled. One of the many adjustments is the replacement of "macros" by defined replacement text. Here we are going to be a little loose with the C specifications, and our sample file is certainly not valid C, but the idea is not very different from what the C preprocessor does.

You must write a Python program that preprocesses the standard input, replacing any macros it sees.

Definition: A *macro* is a string that is to be replaced by a *replacement string*. The macro must be a non-empty string containing no white space. The replacement string must contain at least one non-white-space character, but may also contain white space.

Preprocessing works line by line. Lines that define macros begin with "`#define`", and may appear anywhere in the text of the input. Lines that are not macro definitions are sent to the standard output after macro replacement. Sequences of white-space characters may be replaced by single blank characters if desired.

For example, here is a file "`csource.c`" to be processed:

```
#define Jim, Professor Clarke,
#define Hi Dear
#define HAVETOHAVE would like to ask for
#define ask beseech you
Hi Jim,

I HAVETOHAVE an extension on my project,
#define DRANKTOOMUCH was unwell
because I DRANKTOOMUCH yesterday.
#define yesterday. all week.

#define Cheers, Yours sincerely,
Cheers,
Chris Sam
```

and here is what your program should produce from it:

```
$ python prep.py < csource.c
Dear Professor Clarke,

I would like to ask for an extension on my project,
because I was unwell yesterday.

Yours sincerely,
Chris Sam
```

Notice that some of the macro definitions in the example "source file" have no effect. They are there to answer questions that might occur to you.

This page gives you more space for your answer.

[This blank page is for rough work. This page will **not** *be marked, unless you clearly indicate the part of your work that you want us to mark.]*