15-410, Spring 2007, Homework Assignment 1.
Due Wednesday, February 28, 6:59:59 p.m.

You need only complete 14 of the 24 points.

Note: in an exam setting we would not ask you to write as much assembly code as the first problem.

Please observe the non-standard submission time... As we intend to make solutions available on the web site immediately thereafter for exam-study purposes, please turn your solutions in on time.

Homework must be submitted in either PostScript or PDF format (not: Microsoft Word, Word Perfect, Apple Works, LaTeX, XyWriter, WordStar, etc.). Submit your answers by placing them in the appropriate hand-in directory, e.g., `/afs/cs.cmu.edu/academic/class/15410-s07-users/$USER/hw1/$USER.ps` or `/afs/cs.cmu.edu/academic/class/15410-s07-users/$USER/hw1/$USER.pdf`. A plain text file (.text or .txt) is also acceptable, though it must conform to Unix expectations, meaning lines of no more than 120 characters separated by newline characters (note that this is *not* the Windows convention or the MacOS convention). Please avoid creative filenames such as `hw1/my_15-410_homework.PdF`.

1 Stack Madness (10 pts.)

Doubtless one consequence of working on a thread library is some frustration with how disturbingly *linear* C's stacks are. Let's explore what a different approach might look like.

Stack frames will be allocated as follows. A global array of `void *` named `frametops` will be initialized by the C startup code so that each pointer points to the top four-byte word of a 4-kilobyte page. A global integer named `framenum`, which will count downward, will indicate which frame is currently in use. So the `%esp` values for the current function's stack will range from `frametops[framenum]` down to `frametops[framenum]-4092`. You can assume that `frametops[0]` contains an "illegal pointer" which will generate a page fault—so this stack will eventually refuse to grow downward, as a regular linear stack also does.

The C declarations for the data structures are as follows.

```
void *frametops[N_FRAMES];
int framenum = N_FRAME - 1;
```

1.1 7pts

Present the handful of "function entry" assembly-language instructions necessary to store the calling function's `%esp` in the top word of the callee's new stack page and the caller's `%ebp` in the next word down. Recall that some registers are caller-save and others are callee-save, and don't forget to maintain any necessary invariants. We realize that this setup makes it clumsy to access function parameters (yuck—whose idea was this, anyway?).

1.2 2pts

Present the "function return" instructions necessary to sanely return to the caller.

1.3 1pt

Indicate an important limitation this approach imposes on function behavior.

2 Get to work! (10 pts.)

Consider a real-time system which must respond very quickly to incoming requests from the outside world, though it is acceptable if there is some start-up delay when the system is powered on or at other application-dependent idle times.

We wish to use the "thread group" or "worker thread" abstraction of the Project 2 "thrgrp" library, but wish to add an additional call, `thrgrp_prepare(int n)`, which should ensure that the next `n` invocations of `thrgrp_create()` should start running the first instruction of the indicated `func` "as soon as possible without writing distastefully low-level code."

2.1 9pts

Indicate which data structures you would add to the existing `thrgrp_group_t`. Sketch out what your `thrgrp_prepare()` would do and how you would modify the `thrgrp_create()` we provided you with. You may be brief.

2.2 1pts

If you can think of a desirable modification to `thrgrp_bottom()` or `thrgrp_join()`, explain.

3 Plumbers (4 pts.)

Consider a plumbing company staffed by Alice, Bob, and Cameron. Each worker has an individual "one size fits all" policy for choosing tools, regardless of the job to be accomplished. Each worker's maximum tool usage is as follows.

Worker	Wrenches	Hammers
Alice	2	1
Bob	2	2
Cameron	1	2

3.1 2 pts

Is the following state safe? Why or why not?

Who	Wrenches	Hammers
Alice	1	0
Bob	1	1
Cameron	0	1
Available	1	0

3.2 2 pts

Is the following state safe? Why or why not?

Who	Wrenches	Hammers
Alice	2	1
Bob	2	2
Cameron	0	0
Available	0	0