Homework 4: Fork and schedule

Answer all questions. Submit your hand-written answer sheets to the instructor before the lecture begins.

1 Fork

Execute the following program. `getpid()` returns the `pid` of the current process. `getppid()` returns the `pid` of the parent process.

```
#include <unistd.h>
#include <stdio.h>
#include <sys/wait.h>

int main ()
{
  int pid, i, status;

  printf ("main %d parent %d\n", getpid(), getppid());
  for (i = 0; i < 3; i++) {
    pid = fork ();
    if (pid < 0) {
     printf ("Unable to fork\n");
     return 0;
    }
    if (pid != 0) {
        waitpid (pid, &status, 0);
    }
  }
  printf ("process %d (parent %d) is terminating\n", getpid(), getppid());
  return 0;
}
```

1.1 Turn in:

- The output of the program.
- Draw a tree of the parent-child relationships. A node of the tree contains the `pid` of the process. A directed edge between two nodes represents the

parent-child relationship. E.g., 10 -> 11 means process with `pid` 10 is the parent of the process with `pid` 11.

2 Schedule

Look at the implementation of `schedule` and `switch_threads` routines in Pintos.

2.1 Turn in:

- Write a summary of the `schedule` routine.
- What are the input parameters of `switch_threads`?
- Whose stack `switch_threads` executes on?
- Whose stack `switch_threads` returns on?
- Why `switch_threads` is only saving %ebx, %ebp, %esi, %edi instead of all registers?