Make-up specifications for project 2

First of all, we don't accept late submit (late more than 5 minutes) under any circumstance. Please remember to submit your project before the deadline, even it's a semi-finished product.

The detailed specifications of project 2:

- In Python, we may run your program with the following command:

```
python simulator.py test.asm test.txt test_checkpoints.txt test.in test.out
```

where those files correspond to that mentioned in Project_2.pdf.

- For simplicity, you just need to initialize $fp with the same value as that initialized in $sp, and you can just initialize $gp with the address 32KB above the beginning of the static data section (that is, 0x508000). (for detailed reasons, you may refer to the textbook pages 102-106 and appendix A.5)
- For PC, you should keep PC value to be the first instruction that is not executed yet. (e.g. when you just finish executing the instruction at 0x400004 and dump the register layout out, you should keep PC=0x400008 in your dumped binary file)
- For syscalls 10, 13, 14, 15, 16, 17, you should simulate by directly invoking the Linux APIs (some of them have been discussed in tutorial 4) with the parameters given in the "Arguments" column of the system service table in "MIPS Instruction List". You can just regard syscall 10's behavior to be a normal exit (with status code 0).
- For syscall 9, you need to simulate the program break in your allocated 6MB memory, and make sure to return a pointer to the location in dynamic data so that we can put stuff in.
- For syscalls 5, 8, 12, you just need to read from .in file one line at a time. For syscalls 1, 4, 11, you can just print the argument in the .out file one line at a time.
- You don't need to consider negative numbers for these instructions(div divu mult multu) and syscall(sbrk).
- You don't need to consider overflow exception in instruction add.
- For the data types you need to support, you don't need to consider non-integer numbers(such as float or double) in .word/.byte/.half.
- More test cases are released, combined with the original test case in more_tests.zip. All system calls have been covered in the test cases released, so you may check to see their usages in testing.
- The original fib's register dumps had some issues. The new register dumps had been updated.
- You can just ignore the labels in .data.
- For your report:
  The report of this project should be no longer than 5 pages, and you should not include too many screenshots of your code. In your report, you should write:
  1. Your big picture thoughts and ideas, show us you really understand how are MIPS programs are executed in computers.
  2. The high-level implementation ideas. i.e. how you break down the problem into small problems and the modules you implemented, etc.
  3. The implementation details. i.e. what structures did you define and how are they used.
- Project_2.pdf has been updated, with some typos corrected.