Gebze Technical University
Department Of Computer Engineering
CSE 312 / CSE 504

Operating Systems

Homework #01
Due Date: March 10th 2016
CPU and Process Simulation

In this homework, you will simulate a simple hypothetical CPU with a small instruction set. You will run a number of processes on this CPU and show the memory contents during and after the program execution. The other homeworks (HW2, HW3, ...) will use this CPU architecture, so it is important to complete this homework.

Your CPU (named GTU-C312) has a very small and non-typical instruction set.

Intruction	**Explanation**
`SET B A`	Direct Set : Set the Ath memory location with number B. Example: SET 100, -20 writes the value of -20 to memory location 100.
`CPY A1 A2`	Direct Copy: Copy the content of memory location A1 to memory A2. Example: CPY 100, 120 copies the memory value of address 100 to the memory address 102
`CPYI A1 A2`	Indirect Copy: Copy the memory address indexed by A1 to memory address A2. Example: CPYI 100, 102: if memory address contains 200, then this instruction copies the contents of memory address 200 to memory location 102.
`ADD B A`	Add number B to memory location A
`ADDI A1 A2`	Indirect Add: Add the contents of memory address A2 to address A1.
`JIF A C`	Set the CPU program counter with C if memory location A content is less than or equal to 0
`HLT`	Shuts down the CPU
`SYS PARAMS`	Calls the operting system service with PARAMS, you will not implement this in this homework.
	We may add more instructions for HW2, HW3,

The numbers in the above instruction sets are signed long intergers, which means that ADD 300 -1 can be used as an instruction to decrement the memory location 300 value by 1.
The parameters for the instructions can be memory locations (A, A1, A2), unsigned long numbers (B), or instruction numbers (C).
GTU-C312 does not have any registers, instead it uses special memory locations as registers shown below

Memory Location	Register

0	Program Counter
1	Stack pointer
2-20	Reserved for other uses and other homework

Your program file formats will include two sections: one section for the data, one section for the program instructions. One example program (adds number from 1 to 10 and puts the result in address 51) is as follows

```
#Program sample
Begin Data Section
0 0                #program counter
1 0                # stack pointer
2 0
3 0
4 0
5 0
...
255 0
End Data Section
Begin Instruction Section
0 SET 10 50        # i = 10
1 SET 0 51         # sum = 0
2 ADDI 50 51       # sum = sum + i
3 ADD 50 -1        # i = i - 1
4 JIF 50 6         # Go to 6 if i == 0
5 SET 2 0          # Go to 2 - remember address 0 is the program counter
6 HLT              # end of program, the result is in memory address 51 (sum)
End Instruction Section
```

The data and the instruction section can be as large as needed. You can have comments after the # charater in any line.

When your simulated computer starts running, your BIOS will first read a program file. One example program is above. BIOS will read both data segment and instruction segment. Then it will run a loop similar to

```
while (!myCpu.isHalted())
      myCpu.tick();
```

The CPU function tick will take just one instruction at the Program Counter (PC), execute it, then increment the PC (if no jump). This loop ends when the CPU is halted.

For this homework, you will do the following

- Write a CPU class that can execute the instruction set listed above.
- Write two programs for this sistem.
  - One program sorts N numbers in increasing order. The number N and the numbers are given in the data segment of the program.
  - One program makes a linear search out of N numbers. The number N, the key and the other numbers are given in the data segment of the program.
- Write a simulation program that runs your systems with some command line parameters. There should be parameters for the program name and debug flag.
  - Simulate filename -D 1 : will read the program from filename. In debug mode 1, the contents of the memory will be printed to the screen after each CPU tick (memory address and the content for each adress).
  - In Debug mode 0, the program will be run and the contents of the memory will be

displayed after the CPU halts.

- In Debug mode 2, after each CPU tick, the contents of the memory will be displayed. Your simulation will wait for an from the keyboard and it will continue for the next tick.

We will provide the submissin instructions in a separate document. You should strictly follow these instructions otherwise your submission may not get graded.