3F03 — Midterm Exam I

12 February 2013
11:30–12:20

Do not turn this page until you are told to do so.

Write your name and student number below.

Try to use the space provided on these pages. If necessary, use extra pages.

DO NOT WRITE AFTER 12:20

Problem 1:	_____ /5
Problem 2:	_____ /5
Problem 3:	_____ /10
Problem 4:	_____ /5
Problem 5:	_____ /5

Continued on Page 2

Problem 1 (5 points**)** Write a `Bash` shell script that takes as input a file name, say `filename`, and outputs information about the file (if it exists) as follows

- `filename D`
  if directory
- `filename X`
  if executable
- `filename F`
  if not a directory and if not an executable

If `filename` does not exist, this script should print an appropriate message (you can check if a file exists using `-e`) Also, your script should be able to take any number of arguments.

```
#!/bin/sh
for i
do
  if ! [ -e $i ]
  then
      echo File $i does not exists
      continue
  fi
  if [ -d $i ] # if directory
  then
   echo $i D
  elif [ -x $i ] # if executable
  then
   echo $i X
  else #regular file
   echo $i F
  fi
done
```

Problem 2 (5 points**)** Write a `Bash` shell script that, given a filename, asks you with `Y/N` (yes or no) to delete it. The filename can be a directory name. Also, your script should be able to take any number of arguments. If a file does not exist, it should print an appropriate message.

```
#!/bin/sh
for i
do
if [ ! -e $i ]
then
    echo "File $i does not exists"
```

```
    continue
fi
if [ -d  $i ]  # if directory
  then
    echo "Would you like to remove directory $i [y/n]?"
    read response
    if [ $response == 'y' ]
    then
      echo Removing $i
      rm -rf $i
    fi
else
    echo "Would you like to remove file $i [y/n]?"
    read response
    if [ $response == 'y' ]
    then
        echo Removing $i
        rm -f $i
    fi
  fi
done
```

Problem 3 (10 points) Write an assembly program that reads two integers and stores them in memory in 32 bits. Then it swaps the values at the memory locations and outputs the numbers. Your input and output should look like, e.g.

```
Enter a number   : 3
Enter a number   : 6
First  number is : 6
Second number is : 3
```

Solution

```
%include "asm_io.inc"
segment .data

prompt1 db    "Enter a number  : ", 0
outmsg1 db    "First  number is: ", 0
outmsg2 db    "Second number is: ", 0

segment .bss
input1  resd 1
input2  resd 1

segment .text
```

```
        global  asm_main
asm_main:
        enter   0,0
        pusha
        mov     eax, prompt1
        call    print_string
        call    read_int
        mov     [input1], eax
        mov     eax, prompt
        call    print_string
        call    read_int
        mov     [input2], eax
        mov     eax, [input1]
        mov     ebx, [input2]
        mov     [input1], ebx
        mov     [input2], eax

        mov     eax, outmsg1
        call    print_string
        mov     eax, [input1]
        call    print_int
        call    print_nl
        mov     eax, outmsg2
        call    print_string
        mov     eax, [input2]
        call    print_int
        call    print_nl
        popa
        mov     eax, 0
        leave
        ret
```

Problem 4 (5 points) Suppose that you have in your current directory files `x.c`, `y.c`, `z.c`, and a `makefile` that contains only the line

```
z: x.o y.o z.o
```

If you type `make`, list the commands that will be executed.

```
cc    -c -o z.o z.c
cc    -c -o x.o x.c
cc    -c -o y.o y.c
cc   z.o x.o y.o   -o z
```

Problem 5 (5 points**)** A simple encryption method is two split each byte into two 4-bit nibbles and swap them. For example `01010111` is encrypted as `01110101`. Write a piece of assembly code that encrypts a byte stored in register `AL`. (You do not have to write a complete program, just give the instructions for the encryption.) If you cannot write the assembly code, describe the bitwise operations.

```
mov AH, AL
shl AL, 4
shr AH, 4
or  AL, AH
```