University of Maryland College Park

Dept of Computer Science

CMSC216 Summer 2012

Midterm III

First Name (PRINT): ____

Last Name (PRINT): ____

University ID: ____

I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

Your signature: ____

Instructions

- **Write your name now (we will not wait for you to write your name at the end).**
- You must stop writing once time is over.
- This exam is a closed-book and closed-notes exam.
- Total point value is 100 points.
- The exam is a 50 minutes exam.
- Please use a pencil to complete the exam.
- WRITE NEATLY.

Grader Use Only

#1	Problem 1 (Process Control)	(20)	
#2	Problem 2 (Assembly I)	(15)	
#3	Problem 3 (Assembly II)	(65)	
Total	Total (100)	(100)	

Problem 1 (Process Control) (20 pts)

Define a main function that forks a child process and passes to it the string “mary”. The child process will take the string and return it to the parent after it converts it to uppercase using the toUpper() function provided below. The parent will print the string sends.You can assume system calls will not fail (no need to check for errors). You must use pipes for the exchange of data between the processes.

```
void toUpper(char *data) {
     while (*data != '\0')
         *data++ -= 32;
}
```

Problem 2 (Assembly I) (15 pts)

For all the questions below, you can assume the **stack pointer** has **already being initialized**.

1. (2 pts) Can we have a recursive function that does not require us to save/set the %ebp register?

   a. Yes, it is possible.
   b. No, we always need to save/set the %ebp.
   c. Only if the base case is handled by using %eax.
   d. Only if the result of the function is placed in %eax.
   e. c. and d. above.
   f. None of the above.

2. (5 pts) Add at most **two** assembly code instructions to the code below, so we can have for 4 local variables in the function named **my_function**.

```
my_function:  pushl %ebp
              rrmovl %esp, %ebp
```

3. (4 pts) Without using the %ebp register, add a **single** assembly code instruction to the code below that will place the value of the first parameter into the register %eax.

```
my_second_function: pushl %ebp
                    rrmovl %esp, %ebp
                    irmovl $1, %ecx
                    pushl %ecx
                    pushl %ecx
```

4. (4 pts) Add at most **four** assembly code instructions before the "ret" instruction, so we can correctly return from the function assuming 30 pushl's, 10 popl's and 5 pushl's instructions are executed before the **ret** instruction. You can assume we don't care about the data associated with the pushl's and popl's.

```
my_third_function: pushl %ebp
                   rrmovl %esp, %ebp
                   irmovl $1, %ecx
                   # 30 pushl %ecx instructions executed
                   # 10 popl %ecx instructions executed
                   # 5 pushl %ecx instructions executed

                   # code you should provide goes here



                  ret
```

Problem 3 (Assembly II) (65 pts)

On the next page complete the assembly program that has the functionality associated with C program below. Even though we are using the string “HELLO”, your program should work for any string we provide (do not write a program that only counts characters for the string “HELLO”). **Your solution must be recursive, otherwise you will not receive credit.** You can assume strings are null terminated. Your **instances** function must take two parameters and must not use local variables.

```
#include <stdio.h>

char data[6] = "HELLO";

int instances(char *word, char target) {
   if (word[0] == '\0')  {
      return 0;
   } else {
      if (word[0] == target) {
         return 1 + instances(word + 1, target);
      } else {
         return instances(word + 1, target);
      }
   }
}

int main() {
   char c;

   scanf("%c", &c);
   printf("%d", instances(data, c));

   return 0;
}
```

WRITE YOUR PROGRAM ON THE NEXT PAGE

```
main: irmovl $0x1000, %esp
```

```
.align 4
# HELLO string below
data:
.long 72
.long 69
.long 76
.long 76
.long 79
.long 0
```

<u>DON’T START THE EXAM UNTIL WE TELL YOU SO ☺</u>