

Computer Science 03-60-266 Midterm Examination

Wednesday February 18th 2004

Dr. Alioune Ngom

Last Name:

First Name:

Student Number:

INSTRUCTIONS

- EXAM DURATION IS 2 HOURS.
- OPEN NOTES EXAM: **lecture notes only**. Nothing else allowed.
- READ ALL QUESTIONS CAREFULLY BEFORE ANSWERING.
- Write **all** answers and work in the space provided. Use **no** other paper.
- **WRITE YOUR FINAL ANSWERS WITH A PEN.**
- There are ?? questions (? questions per hour). Answer **all** of them.
- There are ?? pages. Count them before you start.
- Calculators, palms, computers and all other computing devices (excluding fingers and brains) are **not** allowed.
- If you need to make any assumptions, state them clearly with your answers.
- Write your answer neatly. Messy work is very hard to read and may cause you to lose marks.

#1	#2	#3	#4	#5	#6	#7	#8	#9	Total
									100

Q#1. (16 marks) Write a program `ArrayToStack` which copies the first N elements of an array, `Vector`, onto the stack. `Vector` is an array of double-words. Register `ESI` should initially contain the offset address of `Vector`, and register `EBX` contains N . Assume `Vector` and N are already declared, and that N is less than the length of `Vector`. Simply complete the code from `Main:` below.

`Main:`

Q#2. (16 marks) Write a program `StackToArray` which copies back to `Vector` the last N double-word elements pushed onto the stack (by program `ArrayToStack`). Again, `EBX` contains the number N , and `ESI` initially contains the offset address of `Vector`. `StackToArray` should not reverse `Vector` (that is, the elements should return to their initial positions as before the execution of `ArrayToStack`). Start from `Main`: below

`Main`:

Q#3. (16 marks) Using the following data definitions:

```
.data
ArrayB BYTE 10h, 20h, 30h, 40h, 50h
ArrayW Word 100h, 200h, 300h
ArrayD DWORD 10000h, 20000h
```

Find all illegal statements from the following:

1. MOV AX, ArrayB+1
2. MOV AL, ArrayB+2
3. ADD ArrayB, ArrayB+1
4. NEG ArrayW
5. MOV EAX, ArrayW+6
6. MOV ESI, OFFSET ArrayW
MOV EAX, [ESI+6]
7. MOV ESI, 6
MOV EAX, ArrayW[ESI]
8. MOVZX BH, ArrayB
9. MOVSX EBX, ArrayB+2
10. MOV ArrayD+1, EBX

Simply encircle (or check) the number, for each correct instruction

Q#4. (16 marks) Assuming the data segment starts at address 00000000h, fill in the memory map (table below) for the following data definitions:

```
.data
ArrayW Word 1ACDh, -4, 'D'
ArrayB BYTE '543', 2Bh, -14, 'Z'
ArrayD DWORD -5, '5'
```

	----- 0 -----	----- 1 -----	----- 2 -----	----- 3 -----
-- 0000 --				
-- 0004 --				
-- 0008 --				
-- 0012 --				
-- 0016 --				
-- 0020 --				
⋮				

Q#5. (20 marks) Given the following data segment and its corresponding memory map, give the content of EAX, expressed in hexadecimal, when each of the following instruction (or sequence of instructions) is executed:

```
.data
Four LABEL DWORD
ArrayB BYTE -1, 2, -3, 4
Two LABEL WORD
ArrayW Word -5, 6, -7, 8
One LABEL BYTE
ArrayD DWORD -9, 10, -11, 12
```

	----- 0 -----	----- 1 -----	----- 2 -----	----- 3 -----
-- 0000 --	FF	02	FD	04
-- 0004 --	FB	FF	06	00
-- 0008 --	F9	FF	08	00
-- 0012 --	F7	FF	FF	FF
-- 0016 --	0A	00	00	00
-- 0020 --	F5	FF	FF	FF
-- 0024 --	0C	00	00	00

1. MOVZX EAX, [ArrayB+2]

2. MOVZX EAX, WORD PTR ArrayD

3. MOV EAX, -1
MOV AH, BYTE PTR ArrayW+1

4. MOV EAX, Four

5. MOV EAX, [Two+3]

6. LEA EAX, One

7. PUSH Four
PUSH Two
POP EAX

8. MOV EAX, OFFSET ArrayW
PUSH DWORD PTR [One]
POP AH

9. MOV EAX, AB34CD21h
RCR AX, 2

10. MOV ESI, 4
MOV EAX, -1
ADD AH, BYTE PTR ArrayD[ESI]

Q#6. (16 marks) Using bit-wise Boolean operations (AND, OR, XOR, NOT, TEST), and NO arithmetic operations (ADD, SUB, MUL, NEG, CMP, ...), write a program that receives a signed number in the AX register, and alters this (returning answer in AX) as follows using ONLY four bit-wise Boolean operations:

1. If the number is negative, say $-n$, then replace it with the non-negative number $n - 1$, and then
2. If the number is divisible by 8 , then add 6 to it, and then
3. If it is even, add one to it, but if it is odd, subtract one from it.