McGill

October 2018
Midterm Examination

VERSION # D-4

Introduction to Software Systems

COMP 206

October 16th, 6pm

EXAMINER: Dr. David Meger

STUDENT NAME:		**McGILL ID:**									

INSTRUCTIONS:

EXAM:	CLOSED BOOK ☒ OPEN BOOK ☐
	SINGLE-SIDED ☐ PRINTED ON BOTH SIDES OF THE PAGE ☒
	MULTIPLE CHOICE ☒ (FOR QUESTIONS 1-8) NOTE: The Examination Security Monitor Program detects pairs of students with unusually similar answer patterns on multiple-choice exams. Data generated by this program can be used as admissible evidence, either to initiate or corroborate an investigation or a charge of cheating under Section 16 of the Code of Student Conduct and Disciplinary Procedures. ANSWER IN BOOKLET ☐ EXTRA BOOKLETS PERMITTED: YES ☐ NO ☐ ANSWER ON EXAM ☒ (FOR THE 2 LONG ANSWER QUESTIONS #9 and #10)
	SHOULD THE EXAM BE: RETURNED ☒ KEPT BY STUDENT ☐
CRIB SHEETS:	NOT PERMITTED ☐ PERMITTED ☒ Specifications: One 8 1/2X11 double-sided sheet. Can be handwritten or printed.
DICTIONARIES:	TRANSLATION ONLY ☒ REGULAR ☐ NONE ☐
CALCULATORS:	NOT PERMITTED ☒ PERMITTED (Non-Programmable) ☐
ANY SPECIAL INSTRUCTIONS: e.g. molecular models	This examination consists of 8 multiple choice questions and 2 written programming questions. MC answers on the bubble sheet. Write the solution to the programming questions directly in this exam, which must be returned. 60 marks overall: 2 per multiple choice (=16) and 44 for the long answer. You have 60 minutes. Use 1 mark per minute to budget your time. The document should be 10 pages, including the cover page.

Multiple-Choice Questions (2 marks each)

Question 1: In the following command-line, which program(s)'s standard output is(are) displayed to the terminal?

$ ls | sort > sorted.txt

[✓ a)] No standard output is displayed
b) ls
c) sort
d) sorted.txt
e) The standard output of all programs are displayed

Question 2: Assuming a comma-separated value file named "file.csv" with 3 entries for each line. What is a Linux command that prints the middle entry of every line?

a) head –n2 file.csv
b) cat file.csv | grep 2
[✓ c)] cat file.csv | cut –f2 –d,
d) ls –e2 file.csv
e) echo file.csv | cut –f2 –d,

Question 3: What is the binary representation of 142 in a computer's memory when stored as an unsigned char (1 byte)?

a) 1, 100, 10
[✓ b)] 10001110
c) 01110001
d) 11001000
e) 10101010

Question 4: When printing a string in C, the character '\0' indicates:

[✓ a)] The end of the string.
b) A space between words.
c) The program is run on a quantum computer.
d) The end of a line.
e) The pointer cannot be de-referenced safely.

Question 5: What does this BASH script display on the terminal:

```
#!/bin/bash
a=3

for word in fee fi foe fum
do
    a=$(echo $a $word)
   echo $a word
done
```

a) $(echo $a $word) word
   $(echo $a $word) word
   $(echo $a $word) word
   $(echo $a $word) word

[✓ b)] 3 fee word
   3 fee fi word
   3 fee fi foe word
   3 fee fi foe fum word

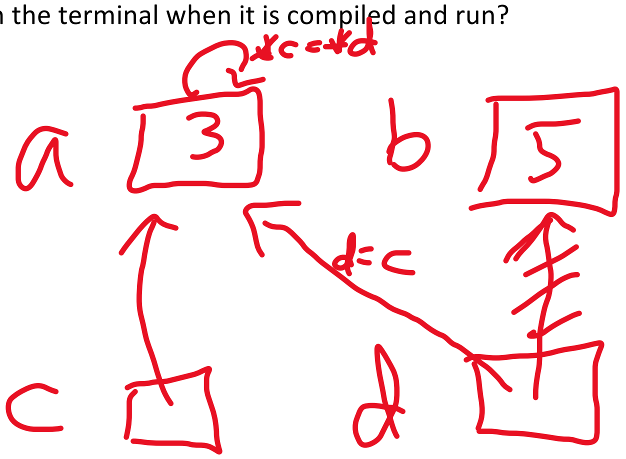
c) a word
   a word
   a word
   a word

d) The program produces an error “Variable not found: word”

e) None of these choices

Question 6: What does the following C program output on the terminal when it is compiled and run?

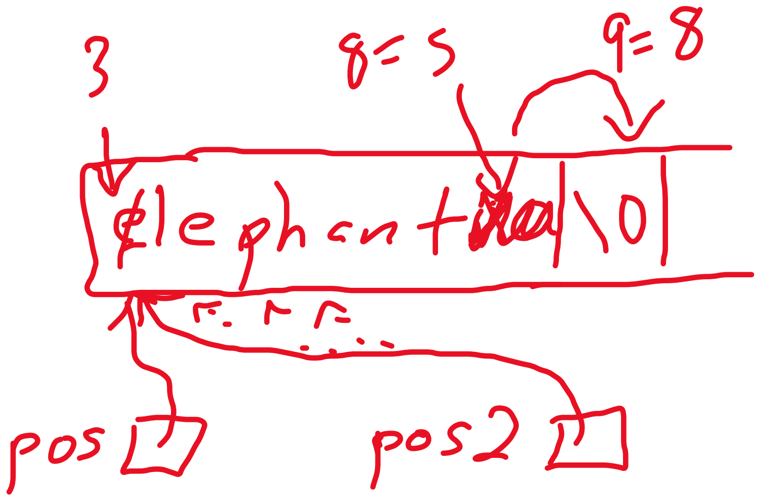
```
#include <stdio.h>

int main( ){
          int a = 3;
          int b = 5;

          int *c = &a;
          int *d = &b;

          d= c;
          *c = *d;

          printf( “a holds: %d and b holds: %d.\n”, a, b );

          return 0;
}
```



a) a holds: 5 and b holds: 3.
b) a holds: 5 and b holds: 5.
c) a holds: 3 and b holds: 3.
d) Segmentation fault
[✓ e) a holds: 3 and b holds: 5.]

Question 7: What does the following C program output to the terminal when compiled and run?

```
#include <stdio.h>

int main(){

        char word[100] = “elephant”;
        char *pos = word;
        char *pos2 = pos;

        *pos += 4;
        pos2++;
        *pos = ‘ ‘;
        pos2++;
        *pos = ‘<‘;
        pos2++;
        *pos = ‘3‘;

        word[9] = word[8];
        word[8] = ‘s’;

        printf( “%s\n”, word );

        return 0;
}
```



a) i <3 ants
b) elephants
c) The output is “elephants” followed by up to 92 random characters
d) Segmentation fault
[✓ e) 3lephants]

Question 8: Which lines contain errors? Assume that they are run in order. Mark only lines where a compilation error is reported, or a crash happens at run-time.

```
 1:   char *ptr = “42”;
 2:   string char[10] = “Gold”;
 3:   int a = ‘\0’;
 4:   int *p = NULL;
 5:   int  b = 3.1415;
 6:   a = *b;
 7:   *p = ‘\0’
 8:   p = &b;
 9:   ptr = *(NULL);
10:   ptr = (char*)p;
```

a) 2, 4, 6, 8
b) 1, 2, 5, 8
[✓ c) 2, 6, 7, 9]
d) 2, 3, 6, 9
e) 3, 5, 7, 10

Written Programming Questions

Question 9: (20 marks) Write a BASH script Named FindFour.bash that takes two arguments:

1) **pattern**: A string holding the pattern to be found.
2) **path**: A path to a directory on the filesystem that should be searched.

Your program must search files within the directory (only one level, not recursive) and print the name of any file that contains **pattern** at least once anywhere on its fourth line. Ignore whether **pattern** occurs additional times or elsewhere in the file. Your program is not allowed to display any other output to the terminal and cannot create any temporary files. Write your FindFour.bash on this page.

```
for f in $2
do
   line=$(cat $f | head-n 4 | tail -n 1)
   if cat $line | grep $1 >/dev/null
   then
      echo $f
   fi
done
```

Question 10: (24 marks) Implement your own version of strcmp. You may use only basic C operations and cannot rely on any libraries (no #include allowed). You will answer by filling in the function that is left empty on the following page. The specification for strcmp is:

```
int strcmp(const char *str1, const char *str2)
```

Compares the string pointed to by **str1** to the string pointed to by **str2**. This function return values that are as follows –

- if Return value < 0 then it indicates str1 is less than str2.
- if Return value > 0 then it indicates str2 is less than str1.
- if Return value = 0 then it indicates str1 is equal to str2.

Here "less" means that the string comes first in alphabetical order (e.g., "abc" < "b"). Note that shorter sub-strings count as "less" (e.g., "ab" > "a"). You can assume that both strings are properly terminated by a null character.

Example

```
int main () {
  char str1[15] = "hello";
  char str2[15] = "jello";
  int ret;
  ret = strcmp(str1, str2);
  if(ret < 0)
     printf("str1 is less than str2");
  else if(ret > 0)
    printf("str2 is less than str1");
  else
    printf("str1 is equal to str2");
  return(0);
}
```

Output:

```
str1 is less than str2
```

Answer to Question 10:

int ~~char *~~ strcmp (const char *str1, const char *str2) {

```
   // Write your strcmp function from here to the closing bracket near the end of the page

     while (*str1){
          int diff = *str2 - *str1,
          if ( diff != 0)
               return diff;

          str1++;
          str2++;
     }

     return *str2 - *str1;
```

} // End of your strcmp function

THIS PAGE INTENTIONALLY BLANK. USE FOR ROUGH WORK IF YOU WISH