SENG 265: Software Development Methods

Fall 2017

Midterm Exam: October 23, 2017

Student name: [redacted]

Student number: [redacted]

Marks: _____

Note:

- There are six (6) sheets of paper in this exam paper containing seven questions. The last question is a BONUS question. Please count the pages in your copy of the exam *and immediately notify the instructor if a page is missing*.
- The exam is out of 90 (ninety) marks.
- All answers are to be written on this paper. You can use the blank pages as scrap space.
- This is a 50-minute exam.

Q1: 4
Q2: 8 8
Q3: 19 DO 19
Q4: 5
Q5: 10 0
Q6: [redacted] 34 post
B: 10

Question 1: Git (4 Marks). 4

Place an X besides all answers that apply.

By stating that git is a version-control system, we are saying:

- It enables a programmer to "lock out" others from accessing a file when that programmer is making changes to the file.
- ✓ X It permits concurrent read access of remote repositories to users who have read-access rights to those repositories.
- ✓ X It tracks changes to file and directories over time.
- X It always automatically resolves conflicting changes when a file has been edited by multiple users.
- None of the above

Question 2: Types in C (10 Marks).

Using one or more typedef statements, define a type MidtermType such that the main function below compiles successfully and produces the output 'Value: 19.6" when run. Any answer that meets those two constraints will be considered correct.

```
#include <stdio.h>
#include <stdlib.h>
/* put your typedefs below */
typedef struct MidtermType {
    MTptr element;
} MidtermType

typedef struct MTptr {
    float* ptr;
} MTptr
```

```
int main() {
    float f;
    f = 19.6;
    MidtermType Q;
    Q.element.ptr = &f;
    printf("Value: %f\n", *(Q.element.ptr));
    Return 0;
}
```

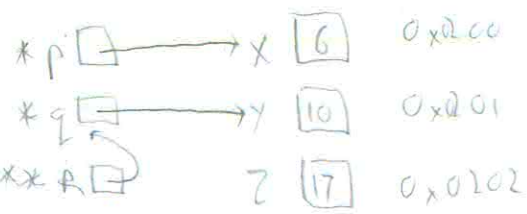
Question 3: Pointers and Addresses (20 marks)

Consider the following syntactically valid C statements, as they would appear in a function:

```
int x,y,z;

int* p;
int* q;
int** R;

x=6;
y=10;
z=17;
p=&x;
q=&y;
R=&q;
```



R = &q
q = 0x0201
p = 0x0200

A. What is the value of the expression "*q - **R"?

(4) Justify your answer:

*q - **R
= *q - *(q)
= *q - *q
= 0

*R = q since R = &q
and **R = *(q) = *q
therefore *q - **R = *q - *q = 0

B. What is the value of the expression "(z -= -- (*q)) + x++"?

(2) Justify your answer:

3

(z -= --(*q))
=> [z = z - (--*q)]
=> [z = z - 9] => z = 17 - 9 = 8

(z -= --(*q)) + x++
=> 8 + 7 ?
= 15

8 + 6
= 14

C. What is the value of z as a result of expression in B?

(4) z = 8

D. Write a single syntactically correct C assignment statement to change the value of the variable y to 0 without using the identifier y.

(4) *q = 0 // *q = y

E. Write a single syntactically correct C assignment statement to change the value of the variable q to point at the variable z without using the identifier q.

(4) *R = &z // *R = q

Question 4: Arrays in C (10 Marks) 5

What is the output of the code below?

```
#include <stdio.h>
#include <string.h>

int main (int argc, char *argv[]) {
      char* array[] = {"January", "February", "April", "June"};
        printf("%c\n", *++array[2]);
      return 0;
}
```

first character only

++array[2] = array[3]

*(++array[2]) = *(array[3]) = "June"

Answer:

Justify your answer.

the out put is "J" and a new line

"J
"

answer p

Question 5: Files and Streams (6 Marks) 10

Consider the syntactically correct C program below, which reads characters from standard input, changes all letters to lowercase (leaving non-alphabetical characters unchanged), then prints each character to standard output.

```
#include <stdio.h>
#include <ctype.h>

int main() {
     FILE* input_file = stdin;
     int c = fgetc(input_file);
     while (c!= EOF) {
          fprintf(stdout, "%c", tolower(c));
          c = fgetc(input_file);
     }
     return 0;
}
```

Cat input |

>

Suppose the program above is compiled into an executable `input_to_lowercase`. Give a valid unix shell command line which uses the `input_to_lowercase` program to change to lowercase every character in a file called `input.txt` and stores the result into a file called `output.txt`.

input_to_lowercase < input.txt > output.txt

% cat input.txt ./input_to_lowercase > output.txt

% ./input_to_lowercase <input.txt > output.txt

Question 6: Strings (40 Marks)

In the space provided below, write an implementation of the C function `remove_vowels` which conforms to the specification below. To get full marks the code should compile (do not use pseudocode).

```
#include <ctype.h>
#include <string.h>

/*remove all vowels (both lowercase and uppercase) from a string s.
        The function must not generate output to either stdout or stderr.
        The letters considered vowels are "A", "E", "I", "O", "U", and "Y".

        Examples:
                If s is "Hello seng265", s will be modified to contain "Hll sng265"
                If s is "Eye", s will be modified to contain "".
                If s is "I am in Seng265", s will be modified to contain " m n Sng265".
*/

void remove_vowels (char* s) {
```

5

BONUS Question (Answer only if you answered all other questions) (10 Marks)

Here are different ways on creating a string in C:

```
char  str1[]    = {'a','b','c','\0'};
char* str2      = "abc";
char  str3[5]   = {'a','b','c','\0'};
char  str4[4]   = "abc";
```

4 bytes
8 byte pointer char*
5 bytes 1 unused
4 bytes 1 unused

On a system with 64-bit (8 byte) pointers, what is the output of the following code?

```
printf("sizeof=%02d \n", sizeof(str1));
printf("sizeof=%02d \n", sizeof(str2));
printf("sizeof=%02d \n", sizeof(str3));
printf("sizeof=%02d \n", sizeof(str4));
```

Answer:
sizeof = 04
sizeof = 08
sizeof = 05
sizeof = 04

(10)