UNITED INTERNATIONAL UNIVERSITY (UIU)

Department of Computer Science and Engineering (CSE)

FINAL EXAMINATION | **TRIMESTER: FALL, 2024**
DURATION: 2 HOURS | **FULL MARKS: 40**

CSE 1111: Structured Programming Language

Answer **all 6 (six)** questions. Figures in the right margin indicate full marks of questions.

[Any examinee found adopting unfair means will be expelled from the trimester/program as per UIU disciplinary rules.]

1. a) Write a C program that satisfies the following requirements. 4

   i. Write a function **`int count_divisors(int n)`** that returns the total number of divisors of a given number `n`.

   ii. Write a function **`int is_perfect_square(int n)`** that returns 1 if the given integer `n` is a perfect square, and 0 otherwise.

   iii. In the **`main`** function, take two positive integers as input from the user. Then determine:
   - which number has more divisors.
   - whether each of the two numbers is a perfect square or not.

   Notes:
   - There are 4 divisors of 6: 1, 2, 3 and 6. So, `count_divisors(6)` should return 4.
   - An integer n is a perfect square if there exists an integer m such that $m \times m = n$.
   - 4 is a perfect square because $4 = 2 \times 2$, but 5 is not a perfect square. So, `is_perfect_square(4)` should return 1 and `is_perfect_square(5)` should return 0.

   b) Determine the output of the following C code. Notice the use of local and global variables in different contexts. 4

```c
#include <stdio.h>

int x = 10;
int y = 20;

int func(int a, int b) {
    int y = 5;
    x += 2;
    y += 3;
    a += y;
    b += x;
    printf("x = %d, y = %d, a = %d, b = %d.\n", x, y, a, b);
    return a * b + x - y;
}

int main() {
    int a = 4, b = 7;
    a = func(a, b);
    printf("a = %d, b = %d, x = %d.\n", a, b, x);
    return 0;
}
```

2. a) Manually trace the following code segment. Show the changes of all the variables. Then write down the output of the program. 4

```
int i, j;
char s1[20] = "ABCd", s2[20] = "EFGH789";

for (i = 0; s1[i] != 0; i++) {
    // do nothing
}

for (j = 0; s2[j] != 0; j++) {
    if (s2[j] >= '0' && s2[j] <= '9') {
        s1[i] = s2[j];
        i++;
    }
}

s1[i] = 0;
printf("S1 = %s", s1);
```

b) Write a function **`void myStrRev(char src[], char dest [])`** that stores the reverse of `src` into `dest`. You are **not allowed** to use **`<string.h>`**. 4

3. a) Determine the output of the following C code segment for each of the given inputs. 2

```
char s[] = "computer club";
char *p, ch;
p = s;

scanf("%c", &ch);

while (*p != ch) {
    printf("%c", *p);
    p++;
}
```

Input:

(i) `u` (ii) `c` (iii) `r`

b) Determine the output of the following C code segment. 2

```
int x, *y;
x = 10;     // address of x: 672454436
y = &x;     // address of y: 672454440

x = x + 1;

printf("%d\n", y);
printf("%d\n", *y);
printf("%d\n", x--);
printf("%d\n", *y);
```

c) Write a program to calculate the factorial of a given number following the following steps. 4

- Input the number in the **`main`** function.
- Pass this number to the **`factorial`** function.
- Calculate the factorial value in the **`factorial`** function.
- Print the factorial result in the **`main`** function. Use the concept of **call by reference**.

4. Write a C program that satisfies the following requirements. $3 + 3 + 2$

   i. Create a structure named `Book` to store details about a book like title, author, and price.

   ii. Input the details of three books and store it in an array of structures.

   iii. Display the information of the book with the highest price and the one with the lowest price.

5. Write down a program that counts number of words in a text file named **`"A.txt"`**. You can safely assume that it's in the same folder as the source code and it doesn't contain any trailing spaces. 4

6. Solve any **one** of the two following problems. 4

   i. Find the bug in the recursive function and rewrite it. The fixed function should work correctly for any positive value of `n`.

```
int addDigits(int n) {
    return (n % 10) + addDigits(n / 10);
}

void main() {
    int k;
    k = addDigits(4531);
    printf("k = %d", k);
}
```

   Expected Output:

```
k = 13
```

   ii. The file **`"input.txt"`** contains 5 integers. Write a program that reads the 5 integers and prints the sum of all even numbers and the sum of all odd numbers in another file **`"output.txt"`**.

   The source code and the text files are in the same folder.

   Examples:

Sample `"input.txt"` *Before* **Running the Program**	**Sample `"output.txt"`** *After* **Running the Program**
`1 2 3 4 5`	`6 9`
`10 15 20 25 30`	`60 40`