Spring Semester 2023

EE209: Programming Structures for Electrical Engineering

Final Exam

Closed book/notes/friends/electronic devices/everything.
Write everything in English.
Write your student ID and name on *every* page.
Hand it in by 3:45 PM
PLEASE MAKE YOUR HANDWRITING LEGIBLE.

Code of Conduct

- All coats and jackets should be placed on the back of each candidate's chair. All notes and books, pencil cases, turned-off cell phones, laptops, and other unauthorized aids, and purses should be stored inside the candidate's knapsack or large bag, which should then be closed securely and placed under the candidate's chair. Candidates are NOT allowed to have a pencil case on their desk; any pencil cases found on desks will be searched. All watches and timepieces on desks will be checked. Candidates are not allowed to touch their knapsack or bag or the contents until the exam is over. Candidates are not allowed to reach into the pockets or any part of their coat or jacket until the exam is over.
- Candidates shall not communicate with one another in any manner whatsoever during the examination. Candidates must stay in the examination room unescorted for any reason, including using the washroom.
- No materials or electronic devices shall be used or viewed during an examination except those authorized by the Chief Presiding Officer or Examiner. Unauthorized materials include, but are not limited to: books, class notes, or aid sheets. Unauthorized electronic devices include, but are not limited to, cellular telephones, laptop computers, tablets, calculators, MP3 players (such as an iPod), Personal Digital Assistants ("PDA" such as a Palm Pilot or Blackberry), electronic dictionaries, Smart Watches and Smart Glasses.
- Candidates who use or view any unauthorized materials or electronic devices while their examination is in progress - or who assist or obtain assistance from other candidates or any unauthorized source – will get the grade F and face possible suspension.
- At the conclusion of an examination, all writing shall cease. The Chief Presiding Officer may seize the papers of candidates who fail to observe this requirement, and a penalty may be imposed.

I have read and adhere to the code of conduct. My signature on this page means my pledge to abide by these standards.

Signature: ____________________________________ Date: ____________________

Name: ____________________________

Student ID: ____________________________

Read the questions carefully and focus your answers on what has been asked. You are allowed to ask the instructor/TAs for help in understanding the questions in case you find them unclear. Be concise and precise in your answers, and state clearly any assumption you may have made. You have 165 minutes (1:00 PM – 3:45 PM) to complete your exam. Be wise in managing your time. Good luck.

Question 1	/ 40
Question 2	/ 15
Question 3	/ 10
Question 4	/ 15
Question 5	/ 15
Question 6	/ 15
Extra	5 points
Total	/

SID:	**Name:**

(40 points) 1. Quick Hits. Please read the questions carefully. You get +2 points for each question if you answer the question correctly. If you do not answer, you get 0 points. If you answer the question incorrectly, you get -2 points.

1.1. $(4000000000)_{10}$ can be represented as int in C.

  a. True
  [✓ b. False]

1.2. Which of the following is the output generated by the print statement in the code segment below?

```
int x = 3;
int *y;
int *z;

y = &x;
z = y;
(*z)--;

printf("result: %d\n", *y + *z);
```

  a. result: 6
  b. result: 5
  [✓ c. result: 4]
  d. None of the above.

1.3. Which of the following statements regarding the rules of the switch construct is TRUE?

  a. Two case labels can have the same constant expression value.
  b. No two case labels can be associated with the same set of actions.
  c. The control expression that follows the keyword switch must be an integral type.
  [✓ d. None of the above.]

1.4. Which of the following statements regarding pointer variables is FALSE?

  [✓ a. Working with an uninitialized pointer variable is a compile-time error.]
  b. The asterisk character (*) is used in two different contexts for pointer variables; for declaration and dereferencing.
  c. The value of a pointer variable can change during the execution of a program.
  d. None of the above.

SID:	**Name:**

1.5. Suppose x is a double. After statements

```
x = 5.9;
a = (int) x;
```

have been executed, what is the value of x?

a. 5.9
b. 5
c. 5.0
d. 6

Grading policy: No partial points.

1.6. What is the output of the printf statement in the code segment below?

```
int x = 0x15213F10 >> 4;
char y = (char) x;
unsigned char z = (unsigned char) x;
printf("%d, %u", y, z);
```

a. -241, 15
b. -15, 241
c. -241, 241
d. -15, 15

Grading policy: No partial points.

1.7. Which of the following statements regarding dynamic memory allocation is TRUE?

a. The result of the malloc function is assigned to an array variable.
b. The value passed to the malloc function is the product of the number of elements to store and the amount of memory required to store a value of the given data type.
c. The memory allocated as a result of the malloc function is initialized to a default value based on the data type specified.
d. None of the above.

Grading policy: No partial points.

1.8. Place parenthesis in the following expression to explicitly show the order of evaluation. For example, a + b * c ⇒ (a + (b * c))

3 + * p ++ ⇒ (3 + (* (p ++)))

Grading policy: No partial points.

SID:	**Name:**

1.9. Which of the following describes the integer value generated by the print statement in the code segment below?

```
int x[5] = {6, 9, 3, 0, 4};
int y[5] = {0};

y = x;

printf("y[0] = %d\n", y[0]);
```

a. The value displayed will be the integer 6.
b. The value displayed will be the memory address represented by the array x.
c. No integer value will be displayed due to a compiler error regarding the assignment statement.
d. None of the above.

1.10. Which of the following statements regarding arrays is FALSE?

a. The memory address represented by the name of an array can change during the execution of a program.
b. When adding an integer to the name of an array, the result is a value corresponding to another index location.
c. The dereference of an array name is the value of its first element.
d. None of the above.

1.11. Which of the following statements regarding index range checking in arrays is FALSE?

a. The issue of attempting to access an index beyond the range of valid index values for an array is a logical error.
b. When attempting to access an index beyond the range of valid index values for an array, it is possible that the output results may not be as expected.
c. The problem of index range checking is limited to only values greater than the largest index values for an array.
d. None of the above.

SID:	**Name:**

1.12. Which of the following statements regarding user-defined functions is FALSE?

a. A variable declared in the local declaration section of a function can have the same identifier as one of the parameters within the same function.

b. Data sent from the calling function to the function being called will be received in the same order in which it was passed.

c. Parameters are defined as local variables in the first line of the function definition and should not be re-defined within the local declaration section of the function.

d. None of the above.

Grading policy: No partial points. (1.12 ~ 1.16 전부 동일)

1.13. Which of the following is the correct ordering (left-to-right) of a file's compilation cycle? (a filename with no extension is an executable):

a. foo.c ⇒ foo.o ⇒ foo.s ⇒ foo

b. foo ⇒ foo.s ⇒ foo.o ⇒ foo.c

c. foo.c ⇒ foo.s ⇒ foo ⇒ foo.o

d. foo.c ⇒ foo.s ⇒ foo.o ⇒ foo

1.14. From the following C declaration, what does f represent?

```
int *(*f[3])();
```

a. an array of pointers to pointers to functions that return int

b. a pointer to an array of functions that return pointers to int

c. a function that returns a pointer to an array of pointers to int

d. an array of pointers to functions that return pointers to int

1.15. Which of the following statements regarding type conversions is FALSE?

a. When the types of two operands for the division operator are different, the lower-ranked type is promoted to the rank of the higher type before the quotient is determined.

b. Explicit type conversion is the programmer taking control and determining the data type of an operand in an expression.

c. In an assignment statement, promotion occurs if the right expression has a lower rank than the variable on the left and demotion occurs if the right expression has a higher rank.

d. None of the above.

1.16. What is the C equivalent of mov %eax,%ecx?

a. eax = ecx

b. ecx = eax

c. eax = *ecx

d. ecx = *eax

SID:	**Name:**

1.17. The OS and the hardware has to manage a single page table in total, regardless of the number of processes.

- a. True
- [✓ b. False]

  Grading policy: No partial points.

1.18. Consider the C declaration

    int array[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};

Suppose that the compiler has placed the variable array in the %ecx register.
How do you move the value at array[3] into the %eax register? Assume
that %ebx is 3.

- a. leal (%ecx,%ebx,4),%eax
- b. leal 4(%ecx,%ebx,1),%eax
- [✓ c. movl (%ecx,%ebx,4),%eax]
- d. movl 8(%ecx,%ebx,2),%eax

  Grading policy: No partial points.

1.19. Which of the following(s) are true?

- [✓ a. malloc() may or may not generate a trap (use system call or system level function).]
- b. malloc() always generates a trap (use system call or system level function)
- c. malloc() always never a trap (use system call or system level function)

  Grading policy: No partial points.

1.20. Which register does not change as a result of the assembly instruction ‘ret’ in x86-64?

- a. %rip
- b. %rsp
- [✓ c. %rax]
- d. None

  Grading policy: No partial points.

SID:	Name:

(15 points) 2. Basic Concepts in Computer Systems

2.1. (2 pt) List at least two examples of possible spatial locality existing in an application's program. Answer in no more than four sentences.

1 pt for each (total 2 pt)
ex) array, instructions, …

2.2. (1 pt) What is a page fault? Answer in no more than two sentences.

1 pt
ex) "The referenced page is not in memory(or page table)"
+) "page(or data, address..)", "memory(or RAM, physical page…)" -> 1 pt
+) correct, but not sufficient -> 0.5 pt

2.3. (3 pt) List three key differences between function calls and exceptions.

1 pt for each (total 3 pt)
ex) Exceptions:

- Processor pushes **additional states onto stack**
- Processor pushes data onto **OS's stack**,not application's stack
- Handler runs in **privileged mode**, not in user mode
- Control sometimes **doesn't return** to next instruction

2.4. (1 pt) What unique property does a "trap" have vs. interrupts/faults/aborts?

1 pt
"Trap is **intentional**ly triggered by a program" -> no partial points

2.5. (1 pt) Briefly explain one key difference between a "program" and a "process".

1 pt
" a process is an instance of a program" -> no partial points

SID:	**Name:**

2.6. (2pts) Name two key abstractions that the process abstraction provides.

1 pt for each (total 2 pt)

1. private control flow (context switch, etc)
2. private address space (virtual memory, etc)

2.7. (1pt) What do you call the flaw/bug in a program where the correctness of the program is critically dependent on the sequence or timing of events?

1 pt. race condition

2.8. (1pt) Which testing method requires more time, statement testing or path testing?

1 pt. path testing

2.9. (1pt) What do you call the testing method that test individual functions rather than the entire system?

1 pt. Unit testing

2.10. (2pts) What is the goal of creating a memory hierarchy in modern computer systems? Fill in the blank.

__high(1pt)___ capacity at the cost of ___low-level(1pt)___ storage

SID:	**Name:**

(10 points) 3. Remove Duplicate in a string (array of char)

You are asked to write a recursive function that removes consecutive duplicates in a string.

For example, if the input string is “aab”, the output string should be “ab”. The function modifies the array of char given as the parameter to write the resulting output string.

For example, when you execute the main() function, it outputs “[10, Helo World]”.
Fill in the blanks below. **Do not use any function calls in your answer.**

```
/* removedup(int len, char s[]) removes consecutive duplicates and returns
the length of resulting string. It stores the output string in s[]. So s[]
is modified */

int removedup(int len, char s[])
{
    int k;
    if (________) return len; // 2 points, no function call


    if (s[0] == s[1]) {
        k= removedup(__________, __________); //no function call

        memmove(s, s+1, k+1);
        return _______; // 2 points, no function call
    }
    k = removedup(__________, __________); //no function call

    return  k+1;
}
int main()
{
    char str[] = "Hello World";
    int k = removedup(strlen(str), str);
    printf("[%d, %s]\n", k, str);
}
```

Note: The C library function void *memmove(void *str1, const void *str2, size_t n) copies n characters from str2 to str1, but for overlapping memory blocks, memmove() is a safer approach than memcpy().

Solution

```
int removedup(int len, char s[])
{
  int k;
        (1)  if (len<=1) return len;
        (2)  if (len<2) return len;
        (3)  if (s[0]==’\0’) return len;
        (4)  if (*s==’\0’) return len;
        (5)  if (s[0]==0) return len;
        (6)  if (!*s) return len;
        **   if (len==1): Wrong (Does not consider inputs with empty string)
        **   if (s[0]==NULL): Wrong (Returns warning on compilation)
  if (s[0]==s[1]) {
    (1)  k = removedup(len-1, s+1);
    (2)  k = removedup(--len, ++s);
    (3)  k = removedup(len-1, &s[1]);
    **   len-- , s++: Wrong (Incorrect functionality)
     memmove(s, s+1, k+1);
    (1) return k;
    ** No other answer
  }
    (1)  k = removedup(len-1,s+1);
    (2)  k = removedup(--len, ++s);
    (3)  k = removedup(len-1, &s[1]);
    **   len-- , s++: Wrong (Incorrect functionality)
  return k+1;
}
```

Grading Policy:

Each blank corresponds to 2, 1.5, 1.5, 2, 1.5, 1.5 points.

Total score is the sum of all scores of the blanks.

No partial points are given within a blank.

SID:	**Name:**

(15 points) 4. main() function and assembly

The main function has two arguments:

`int main(int argc, char** argv)`

Below is the x86-64 assembly code that implements the main function.

```
1           .global main
2           .text
3    main:
4           push    %rdi                    # save registers
5           push    %rsi
6           sub     $8, %rsp                # must align stack before call

8           mov     (%rsi), %rdi            # the argument string to display
9           call    puts                    # print it

10          add     $8, %rsp              # restore %rsp to pre-aligned value
11          pop     %rsi                  # restore registers
12          pop     %rdi

            add     $8, %rsi                 # point to next argument
            dec     %rdi                     # count down
            jnz     main                     # if not done countinue

            ret
```

Note puts() writes the string s and a trailing newline to stdout.

4.1. (5 pts) What might happen if lines 4-5 and lines 11-12 are removed? In other words, why do we need to save the registers?

Answer:
We have to save(lines 4-5) and restore (lines11-12) the registers (i.e., %rdi and %rsi) as they can be modified by the function named 'puts'.

Grading policy:
- 5points if the answer includes the contents that "the registers can be modified by the callee function (put)".
- 3points if the answer only includes the keyword "caller-saved register".
- 1point if the answer includes the contents that "system crashes".

SID:	Name:

4.2. (5 pts) Assume you compiled the program and the program is named ee209. Write the output of:
`./ee209 what is the output of the program?`
Use one row for each line of output. You can leave the remaining rows blank if the output is shorter than the number of the provided rows. The numbers on the left are line numbers.

1	**./ee209**
2	**what**
3	**is**
4	**the**
5	**output**
6	**of**
7	**the**
8	**program?**
9	
10	

Grading criteria

+5: every line is correct
-1 for each minor mistake. e.g.) spelling, missing "?"
+0: except all (including missing ./ee209)

4.3. (5 pts) Describe in one sentence what the function does.

Answer) print every command-line argument line-by-line.

Grading criteria

+5: correct answer
+0: wrong answer. e.g.) prints "the input string"

SID:	Name:

(15 points) 5. Assembly to C

The following problem tests your understanding of switch statements that use jump tables.

Consider a switch statement with the following implementation. The code uses this `jmpq` instruction to index into the jump table:

```
0x40047b jmpq *0x400598(,%rdi,8)
```

Using GDB we extract the jump table:

```
0x400598:   0x0000000000400488      0x0000000000400488
0x4005a8:   0x000000000040048b      0x0000000000400493
0x4005b8:   0x000000000040049a      0x0000000000400482
0x4005c8:   0x000000000040049a      0x0000000000400498
```

Here is the assembly code for the switch statement:

```
#on entry : %rdx = c and %rsi = b
0x400474 :  cmp   $0x7,%edi
0x400477 :  ja    0x40049a
0x400479 :  mov   %edi,%edi
0x40047b :  jmpq  *0x400598(,%rdi,8)
0x400482 :  mov   $0x15213,%eax
0x400487 :  retq
0x400488 :  sub   $0x5,%edx
0x40048b :  lea   0x0(,%rdx,4),%eax
0x400492 :  retq
0x400493 :  mov   $0x2,%edx
0x400498 :  and   %edx,%esi
0x40049a :  lea   0x4(%rsi),%eax
0x40049d :  retq
```

Fill in the C code implementing this switch statement (fill the underlined):

```
int main(int a, int b, int c){
      int result = 4;

      switch(a){
            case 0:
            case 1:
                  c = c - 5;

            case 2:
                  result = 4 * c; //or result *= c
                  break;

            case 5:
                  result = 86547; //or 0x15213
                  break;

            case 3:
                  c = 2;

            case 7:
                  b = b & c;

            default:
                  result += b; // or result = b + 4
      }

      return result;
}
```

[Grading Policy : 15pt total, point for each line follows:

2pt	**c = c - 5**	
1.5pt	**case 2**	**case 5**
2pt	**result = 4 * c //or result *= c**	**result = 86547 //or result = 0x15213**
1.5pt	**case 5**	**case 2**
2pt	**86547 //or 0x15213**	**4*c //or *=c**
2pt	**c = 2**	
2pt	**b = b & c**	
2pt	**result += b //or result = b + 4**	

No partial point for each line.]

SID:	**Name:**

(15 points) 6. Process, signal, and I/O

6.1. (5pts) fork() **[Grading Policy : Each 2.5point, if the order is correct, 1point]**

How many lines of output does the following functions generate? Assume the program compiles and runs without any error. Give your answer as a function of n. Assume n≥1.

```
void foo1(int n) {
    int i;
    for (i=0; i<n; i++) {
        fork();
    }
    printf("Hi\n");
}
```

Number of lines of output :

Ans) 2^n

```
void foo2(int n) {
    int i;
    fork();
    for (i=0; i<n; i++) {
         printf("Hi\n");
    }
 }
```

Number of lines of output :

Ans) 2n

6.2. (5pts) I/O. Consider the following code. Assume all system calls succeed, and that calls to read() and write() are atomic with respect to each other. **[Grading Policy : All or Nothing]**

The contents of foo.txt are "12345".

What is the output of the following program?

Answer: ______________________________

```
void read_and_print_one(int fd)
```

```
{
    char c;
    read(fd, &c, 1);
    printf("%c", c);
    fflush(stdout);
}
int main(int argc, char *argv[])
{
    int fd1 = open("foo.txt", O_RDONLY);
    int fd2 = open("foo.txt", O_RDONLY);
    read_and_print_one(fd1);
    read_and_print_one(fd2);
    if (fork()==0) {
        read_and_print_one(fd2);
        read_and_print_one(fd2);
        close(fd2);

        fd2 = dup(fd1);
        read_and_print_one(fd2);
    } else {
        wait(NULL);
        read_and_print_one(fd2);
        printf("\n");
    }
    close(fd1);
    close(fd2);
    return 0;
}
```

Solution: 112324

6.3. Signal (5pts) **[Grading Policy : All or Nothing]**

Consider the following C program. The child sends a user-defined signal SIGUSR1 to the parent after completing some work. Please fill in the blank to send SIGUSR1 to the parent.

```
int counter = 0;
void handler (int sig) {
    counter++;
}
int main() {
    signal(SIGUSR1, handler);
    int parent = getpid();
    int child = fork();
    if (child == 0) {
        // omitted: assume we did some work here
        kill(_________, ____________);
        exit(0);
    }
    sleep(1);
    waitpid(child, NULL, 0); /* wait for the completion of child */
    return 0;
}
```

kill (parent, SIGUSR1);

(Extra 5 points) Do not write your name on this sheet of paper.

Which of the following describes your situation best?

(a) Before taking EE209, I had a strong interest in computers and now I have even more interest. I plan to take more computer courses and if possible, find a job or go to graduate school in the field of computers.
(b) Before taking EE209, I had a strong interest in computers but now I have less interest in computers.
(c) Before taking EE209, I had little interest in computers but now I have more interest. I plan to take more computer courses and if possible, find a job or go to graduate school in the field of computers.
(d) Before taking EE209, I had little interest in computers and I still do.
(e) None of the above (please elaborate).

Please explain and describe your situation in detail. We are asking for your opinion so that we can improve the course in the future. Thanks!