Faculty of Computer Science

DALHOUSIE UNIVERSITY
Inspiring Minds

Final Exam

Term: Fall 2018 (Sep4-Dec4)

Student ID Information

Last name: ______________ **First name:** ______________

Student ID #: ______________ **CS.Dal.Ca userid:** ______________

Course ID:	**CSCI 2132**
Course Title:	Software Development
Instructor:	Vlado Keselj
Date of Exam:	9 Dec 2017
Time Period: Start:	12:00
End:	15:00
Duration of Exam:	3 hours
Exam Location:	Dalplex
Number of Exam Pages: (including this cover sheet)	21 pages
Exam Type:	**Closed Book**
Additional Materials Allowed:	**One letter-format paper (8.5”×11”) with anything written or printed on it (both sides). No textbooks, computers, calculators, or other aids are allowed.**

Grade Table	
Question	Score
1	/12
2	/12
3	/6
4	/6
5	/7
6	/9
7	/8
8	/8
9	/10
10	/10
11	/10
12	/9
13	/8
14	/32
Σ	/147

1. (12 points) True-false questions. 2 points each. No justification necessary, but it may be helpful if the question seems ambiguous.

a) (2 points) In the UNIX file model, a file is a stream of bytes.

b) (2 points) Symbolic links and pipes are types of files in Unix.

c) (2 points) In the C programming language, the following two pairs of `scanf` format strings are equivalent:

```
"%c,%c"
"%c ,%c"
```

d) (2 points) In the C programming language, the following two pairs of `scanf` format strings are equivalent:

```
"%d,%d"
"%d ,%d"
```

e) (2 points) If there is a text file named `log.txt` in the current working directory, the command `date >> log.txt` will append the output of the `date` command to the end of `log.txt`.

f) (2 points) In a shell script, whenever we used the command `exit` to terminate the script, we are always required to provide an exit code explicitly as the argument of the `exit` command.

2. (12 points) Multiple-choice. Circle the *single* best answer.

a) (3 points) Which of the following statements regarding streams and files in the C programming language is INCORRECT?

A. In C, a stream is any source of input or any destination of output;

B. The following statement will output “out of memory” followed by a newline character to the standard error channel:
`fprintf(stdout, "out of memory\n");`

C. The `fopen` function returns NULL when it fails to open a file;

D. The notion of lines does NOT apply to binary files.

---

b) (3 points) If `c` is a variable of type `char`, which one of the following statements is illegal?

A. `i += c;  /* i has type int */`

B. `c = 2 * c - 1;`

C. `printf(c);`

D. `putchar(c);`

---

c) (3 points) Suppose that the following declarations are in effect:

```
int a[] = {17, 211, 10, -5, 14, 19};
int *p = &a[3];
```

What is the value of `*(p+2)`?

A. 10

B. -5

C. 14

D. 19

---

d) (3 points) A file `names` contains people names. Each line contains a first name and a last name of a person (e.g. `John Smith`), and the file is not sorted. Our task is to list only first names, sorted alphabetically, and no name should be repeated in more than one line.

Which one of the following four command will complete this task correctly?

A. `cut -d " " -f 1 < names | uniq | sort`

B. `cat names | cut -d " " -f 1 | sort | uniq`

C. `sort | uniq | cut -d " " -f 1 names`

D. `sort < names | uniq | cut -d " " -f 1`

3. (6 points) Program Output.

What is the output of the following program? You can justify your answer, but it is not required.

```
int a[10] = {1}, i;
int *p = &a[4];
int *q = p+1;

for (i=1; i<10; i++) a[i] += a[i-1]+2;

*(p++) = *(q++); q++; *(p++) = *(q++);

printf(" %d %d %d\n", *a + *q - *p, q-p, *q-*p);

for (i = 4; i < 8; i++) printf(" %d", a[i]);
```

4. (6 points) Explain briefly.

Show the content of the standard `main` parameters: `argc` and `argv` when we call the compiled program in the following way:

```
./a.out apple pine orange cherry
```

If there are any pointers or strings, show them graphically.

5. (7 points) Program Output. What is the output of the following program?

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

int main() {
  char s[90] = "test";
  char *p = "program";

  printf("%d %d\n", strlen(s), strlen(p+2));
  puts(strcat(s, p));
  printf("%d\n", strlen(s));

  return 0;
}
```

6. (9 points) Single command line. For each of the following questions, write a single Unix command line to perform the task required.

a) (3 points) Print a list of files in the directory `/usr/bin` whose names end with the English word "make". The file named `make` is considered one of these files.

b) (3 points) Print out the number of six-character words in the Linux dictionary file `/usr/share/dict/linux.words` starting with `a` or `c` and ending with `h`.

Recall that this dictionary file contains one English word per line.

c) (3 points) Print out a list of regular files in the directory `/usr/bin` whose file owner has read permission but does not have execute permission.

7. (8 points) Give concise answers.

a) (4 points) Write down the full names (NOT abbreviations) of four `gdb` commands, i.e. those commands that you can enter in the `gdb` console. For each command you put down, write a short sentence to explain what this command is for. You are not required to explain the usage such as parameters or options of these commands; the explanation that you write just have to be sufficient to show that you know what these commands do.

b) (4 points) Suppose that you are working in a directory that is a working copy of a directory in SVN. You created a new file, named `file.c` in your directory. Which command lines are required to save this file in the SVN repository?

8. (8 points) Give concise answers.

a) (4 points) Briefly explain the keyword `static` when used with a variable inside a function and outside a function.

b) (4 points) The following is supposed to be an implementation of the `strcpy` function in the standard C library, but there are two errors. Find out these errors, explain why they are incorrect, and fix these errors by modifying the code printed below.

```
char* strcpy(char *s1, const char *s2) {
  int i = 0;

  while (s2[i] != '\0') {
    s1[i] = s2[i];
    i++;
  }

  return s2;
}
```

9. (10 points) Large program organization.

(10 points) Assume that you have a project with the following files: `main.c` containing the main function, `llist.c` containing a linked list implementation, and `llist.h` a header file containing all function prototypes. The file `llist.h` is included in the two source files `main.c` and `llist.c`.

a) (4 points) What code you need to write in `llist.h` to protect it against double-inclusion?

b) (6 points) Write a makefile to compile the program into the executable named `llist` when we run the command '`make`' or '`make all`'. The make should do separate compilation of files `main.c` and `llist.c`.

10. (10 points) Give brief answer.

a) (5 points) Briefly describe what the `malloc` function does and its time efficiency.

b) (5 points) Briefly describe the `mergesort` algorithm, its advantages and disadvantages, and discuss running time. If you want to use code, use pseudo code.

11. (10 points) Code snippets

(10 points) Write a C program that reads a sequence of integers, one integer per line, from the standard input. For each input integer n, such that $n \geq 1$, the program must print the sum of all positive integers k such that $1 \leq k \leq n$. The input will contain one integer less than 1, and the program must terminate once it reads this integer.

12. (9 points) Shell scripting

a) (3 points) Briefly explain how shell interprets double parentheses in a command, such as: `((` *some expression* `))`

b) (6 points) Briefly explain the lines (1), (2) and (3) in the shell script below:

```
if [ ! -d $1 ]; then
   echo Error                              (1)
   exit 1
fi

if [ -f $1/testfile.txt ]; then            (2)
  if [ ! -e tmp/testfile.txt ]; then       (3)
      cp $1/testfile.txt tmp/testfile.txt
  fi
fi
```

13. (8 points) File operations in C

(8 points) Consider the following two choices for opening a file:

```
FILE *fp = fopen("foo", "ab+");  /* choice A */
FILE *fp = fopen("foo", "a+");   /* choice B */
```

and the following two choices for writing the `double` variable `x` to the file:

```
fprintf(fp, "%lf\n", x);            /* choice C */
fwrite(&x, sizeof(double), 1, fp);  /* choice D */
```

a) Which choices for opening the file and writing to the file are appropriate to be used together? Why?
b) Give one advantage for each choice of writing `x` to the file.

14. (32 points) Write a C program.

(32 points) Assume that you are working on a program to process product prices. The data will be stored in a linked list where the structure of a node is given in C as follows:

```
struct node {
  char prod[50];      /* prod  is product name */
  double price;       /* price is product price */
  struct node *next;  /* next  is pointer to next node */
};
```

(a) (4 marks) Write a C function `printprod` that takes a pointer to the above node structure and prints a line with the following: the product name, a comma (`,`), the product price, another comma, and a newline character to the standard output. An output example is: `office desk, 356.78,`
The function prototype must be: `void printprod(struct node *n);`

(b) (10 marks) Write the C function `readprod` which allocates a node structure, and fills it with a product name, price, and sets member `next` to NULL. The product name and price are read from the standard input. The input format is as follows:
`orange juice, 5.95, ....`
so, the product name consists of some characters that are not comma, followed by a comma, followed by product price, followed by comma, and some other optional characters in a line. You can assume that product name is not longer than 49, and the function should also read all characters to the end of line but not store them anywhere. You do not need to do much error checking except that if you cannot read input in required format the function should return NULL. The function returns a pointer to the structure created like this. The function prototype is: `struct node* readprod();`

(c) (10 marks) Write the C function `insert` which inserts a node into a linked list. You can assume that list has products sorted by price from high to low, and after inserting the node, the list should remain sorted. The first argument `head` is the pointer to the first element of the list, or NULL if the list is empty. The second argument `newnode` is the new node to be inserted. The function returns the new head of the list. The function prototype is: `struct node* insert(struct node *head, struct node *newnode);`

(d) (8 marks) Write a C program that reads a sequence of products and prints them out sorted by their price from high to low. The program must read the the products using the function `readprod` defined in (b), insert them into a linked list using the function `insert` defined in (c). The products must be printed using the function `printprod` defined in (a). Write a complete program with includes and the function main, but you can assume that the functions `printprod`, `readprod`, and `insert` are already defined and their prototypes included, and `struct node` already defined.

An extra page for the last question.

(extra blank page 1 provided)

(extra blank page 2 provided)