Université d'Ottawa
Faculté de génie

École d'ingénierie et de
technologie de l'information

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Information
Technology and Engineering

GNG1106/1506
Final Examination
(Solution)
December 15th, 2008

Time allowed: 3 hours
Closed book examination
Non-programmable calculators are allowed

Attempt all questions
Questions carry the weights indicated
The total number of points for the examination is 50

Answer the questions in the spaces provided
Use both sides of these sheets if necessary

Name: _____ **Student Number:** _____

Part 1:	20	
Part 2:	10	
Part 3:	20	
Total:	50	

⇐ Do not write in this box!

Number of pages: 11

Part 1 – Short Answer Questions

(a) What is the main difference between a C array and a C structure? (2 points).

A C array is a collection of values all of the same type which C structures is a collection of values of different types.

(b) Which of the following is a correct define directive? (1 point)

```
define speed of light 30000
#define LONG_NUMBER 12345678901234567890
#define SHORT 0.01;
#DEFINE RADIUS 30
```

(c) Write the C declaration of variable *str* which is a pointer to a character string. (1 point)

```
char *str;
```

(d) Define a C structure named *student* with the following members: *studentNumber*, *studentName*, *grade*, and *courseCode*. Define a new type *STUDENT* that represents the C *student* structure. (2 points)

C Code

```
struct student
{
   unsigned int strudentNumber;
   char studentName[50];
   double grade;
   char courseCode[20];
};
typedef struct student STUDENT;
```

(e) Provide the output of the following C code in the adjacent box. (2 points)

```
# define COL 3
# define ROW 4

int main( )
{
   int i, j;

   for (i=ROW; i>0; i=i-1)
   {
       for (j=0; j<i; j=j+1)
           printf("*");
       printf("\n");
   }
}
```

Output Screen

```
****
***
**
*
```

(f) Write a C function `swap()` that receives two addresses (to two integer values) in its parameters (two pointers) and swaps the values pointed to by the pointers. The function should have a type `void` (i.e. returns no value). (2 points)

C Code

```
void swap(int *n1, int *n2)
{
   int temp;
   temp = *n2;
   *n2 = *n1;
   *n1 = temp;
}
```

(g) Provide the C instruction(s), in the box below, for the following pseudo-code: (2 point)

If string1 is the same as string2
Print "User authenticated"
Otherwise
Print "Not authenticated"

C Code

```
if(strcmp(string1, string2) == 0 )
   printf("User authenticated");
else
   printf("Not authenticated");
```

(h) Provide the output of the following C code in the adjacent box? (2 points)

```
main( )
{
   int pecans,apples;
   pecans = 100;
   apples = 101;
   printf("Values are %d %d\n",
           pecans, apples);
   fixup(pecans,&apples);
   printf("Now, they are %d %d\n",
           pecans, apples);
}

void fixup(int nuts, int *fruit)
{
   nuts = 135;
   *fruit = 172;
}
```

Output Screen

```
Values are 100 101
Now, they are 100 172
```

(i) Write C code that checks if a file, named "`GNG.dat`" exists or not. Print "File exists" if the file exists, and "File does not exist" otherwise. Be sure **not** to leave any open files after the code has executed and the file data should **not** be altered or deleted (Assume that a file pointer, `fptr`, is already declared). (2 points)

C Code

```
fptr = fopen("GNG.dat", "r");
if(fptr == NULL)
   printf("File does not exist");
else
{
   printf("File exists");
   fclose(fptr);
}
```

(j) Provide Visual Basic for Applications code (that runs in the Microsoft Excel application) to

- Assign a random integer number between 0 and 100 to the cell (2, 3) in the third sheet of an Excel file. (2 points)

```
Worksheets(3).cell(2,3) = Int(101*Rnd())
```

- Read 10 integer values from the first column of the first Excel sheet (cells A1 to A10) and store them in the array *dataArr* (use a For/Next loop structure). (2 points)

```
Dim dataArr(1 to 10) as Integer
Dim rix as Integer
For rix = 1 to 10
      dataArr(rix) = Worksheets(1).cell(rix, 1)
Next
```

Part 2 – Computing Concepts (10 points)

For the program on the following page:

(a) Show how the contents of the working memory are changed by the execution of the program. Record successive assignments to a variable/parameter as follows:

Variable name	~~2~~, ~~6~~, ~~4~~, 10

(b) In the case of character arrays, show all characters in the arrays (even those after the nul character) and the changes made to the contents of the array as follows:

Array name	‘a’ ‘b’ ‘c’ ‘d’ ‘~~e~~’ ‘f’ ‘g’ ‘\0’ ‘\0 ‘\0’ ‘\0’

(c) On the terminal screen, show the output of the program, i.e., what is printed on the screen.

Note that the numbers in parentheses give addresses of memory locations.

Program Memory

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define LEN 10
void print1(char *, int );
void print2(char [][], int, int);

/*-----------------------------------------------*/
int main( )
{
   char days[7][LEN] = {"Monday","Tuesday","Wednesday",
            "Thursday","Friday","Saturday","Sunday"};

   print1(days[1], 4);
   print2(days, 4, 3);
   system("pause");
   return (0);
}

/*-----------------------------------------------*/
void print1(char *ptr, int size)
{
   char day[LEN]= "";

   strcpy(day, ptr);
   *(day+size) = '\0';
   puts(day);
}

/*-----------------------------------------------*/
void print2(char ptr[][LEN], int index, int size)
{
   char day[LEN]= "";

   strcpy(day, ptr[index]);
   *(day+size) = '\0';
   puts(day);
}
/*-----------------------------------------------*/
```

Terminal/Output Screen

Tues
Fri

1 point

Working Memory

days

(10222)	'M' 'o' 'n' 'd' 'a' 'y' '\0' '\0' '\0' '\0'
(10232)	'T' 'u' 'e' 's' 'd' 'a' 'y' '\0' '\0' '\0'
(10242)	'W' 'e' 'd' 'n' 'e' 's' 'd' 'a' 'y' '\0'
(10252)	'T' 'h' 'u' 'r' 's' 'd' 'a' 'y' '\0' '\0'
(10262)	'F' 'r' 'i' 'd' 'a' 'y' '\0' '\0' '\0' '\0'
(10272)	'S' 'a' 't' 'u' 'r' 'd' 'a' 'y' '\0' '\0'
(10282)	'S' 'u' 'n' 'd' 'a' 'y' '\0' '\0' '\0' '\0'

ptr **10232** 1 point

size **4** 1 point

day **'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' 'T' 'u' 'e' 's' 'd' 'a' 'y' '\0' '\0' '\0' '\0'** 2 points

ptr **10222** 1 point

index **4** 1 point

size **3** 1 point

day **'\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' 'F' 'r' 'i' 'd' 'a' 'y' '\0'** 2 points

CPU

Part 3 –Problem Solving (20 Points).

A battery production company is creating software to translate measurement data for 12V DC batteries into a set of specifications. Measurement data for each battery is stored in a binary file in separate structure values. A software program must be developed to repeat the following for each structure value in the binary file:

- read the structure value from the file into a structure variable,
- complete the battery specifications using the measurement data in the structure variable;
- store new specifications in the structure variable,
- saved the updated structure variable contents into another file.

Please review the partial software report provided in the exam annex (focus mainly on understanding the information in Steps 1 and 2 of the report). Then read each of the following questions carefully and return to the annex to obtain the necessary information to answer each question.

Question 3a: Test cases (3 points)

Provide the objectives of three different test cases to test the software. You do NOT need to provide a detailed presentation of each test case; just provide a short statement on the objective of the test cases you are proposing. Ensure that each of the three test cases test a different aspect of the software.

- **Test the proper translations for a number valid entries (5 to 10 battery specifications).**
- **Test for bad input file name given by user.**
- **Test for existing output file, do not overwrite.**

1 point for each suitable test case objective for a total of 3 points

Question 3b (6+3 points): Translate the provided pseudo-code for sub-programs *openOutputFile* and *computeRatings* to C functions.

```
openOutputFile( )
        Repeat
                Print "Please give the name of the output file: "
                Read string into fileName
                Open fileName for reading with pointer fPtr
                If fPtr does not equal NULL
                        Close file with pointer fPtr
                        Print "File ", fileName, " exists - do you wish to overwrite (y/n)"
                        Read character into answer
                        If answer equals 'y'
                                Open file fileName for writing with pointer fPtr
                                If fPtr equals NULL
                                        Print "Cannot open file ", fileName, " for writing"
                        Otherwise
                                Assign NULL to fPtr
                Otherwise
                        Open fileName for writing with pointer fPtr
                        If fPtr equals NULL
                                Print "Cannot open file ", fileName, " for writing"
        While fPtr equals NULL
        Return fPtr
```

```
/*-----------------------------------------------
Function: openOutputFile
-----------------------------------------------*/
FILE *openOutputFile()
{
    FILE *fPtr;
    char fileName[STRINGSIZE];
    char answer;
    do
    {
         printf("Please give the name of the output file: ");
         scanf("%s",fileName);
         fflush(stdin);
         fPtr = fopen(fileName,"rb");
         if(fPtr != NULL)
         {
            fclose(fPtr);
            printf("File %s exists - do you wish to overwrite (y/n):\n", fileName);
            scanf("%c",&answer);
            fflush(stdin);
            if(answer == 'y')
            {
                fPtr = fopen(fileName,"wb");
                if(fPtr == NULL)
                   printf("Cannot open %s for writing\n", fileName);
            }
            else fPtr = NULL;
         }
         else
         {
             fPtr = fopen(fileName,"w");
             if(fPtr == NULL)
                printf("Cannot open %s for writing\n", fileName);
         }
    }
    while(fPtr == NULL);
    return(fPtr);
}
```

1 point for declarations

0.5 point for do/while

1 point for reading in the file name (3 statements)

0.5 point for opening file and if statement

1 point closing file, prompting and reading answer (4 statements)

0.5 point for if statement (includes else instruction bloc)

0.5 point for opening file for writing (includes if statement)

0.5 point for opening file for writing (includes if statement)

0.5 point for return statement

computeRatings(specsPtr)

computePeukert(address of dMeasures referenced by specsPtr,
address of n, address of cp)

Assign computeCapRating(n, cp, 5) to c5hr referenced by specsPtr

Assign computeCapRating(n, cp, 20) to c20hr referenced by specsPtr

Assign computeRC(n, cp) to rc referenced by specsPtr

```
/*-----------------------------------------------
Function: computeRatings
Description: This function uses the contents of the member dMeasures
             structure variable to update the c5hr, c20hr and rc members. It
             calls computePeukert to obtain values of n and Cp  It uses these
             values in calls to computeCapRating for updating c5hr and c20hr,
             and to computeRC for updating rc.
-----------------------------------------------*/
void computeRatings(SPECS *specsPtr)
{
     double n;
     double cp;
     computePeukert(&specsPtr->dMeasures, &n, &cp);
     specsPtr->c5hr = computeCapRating(n, cp, 5);
     specsPtr->c20hr = computeCapRating(n, cp, 20);
     specsPtr->rc = computeRC(n, cp);
}
```

1 point for declarations

0.5 points for each call for a total of 2 points

Question 3c (5+3 points): Provide the pseudo-code for the sub-programs *openInputFile* and *computerPeukert*. DO NOT give C code.

❖ *openInputFile*: This sub program prompts the user for the name of the input file (complete name including extension). The user is prompted until an existing file can be opened.

```
openInputFile( )
        Repeat
                Print "Please give the name of the input file: "
                Read string into fileName
                Open file fileName for reading with pointer fPtr
                If fPtr equals NULL
                        Print "File ", fileName, " does not exist"
        While fPtr equals NULL
        Return fPtr
```

1.5 point for loop statement

0.5 point for prompt

1 point reading string into character array

1 point for opening file and decision statement to check results of the open operation (the if statement is optional and not required).

1 point for return statement

❖ *computePeukert*: With the values reference by a pointer to a DISCHARGE structure value, the values for Peukert's exponent *n*, and Peukert's capacity *cp* are calculated.

```
computePeukert(mPtr, nPtr, cpPtr)
        Assign log(t2 referenced mPtr / t1 referenced by mPtr) /
                (log(i1 referenced by mPtr)-log(i2 referenced by mPtr)
                to content pointed by nPtr
        Assign (t1 referenced by mPtr) (i1 referenced by mPtr)content pointed by nPtr
                to content pointed by cPtr
```

1.5 points for assignment (or equivalent number of statements)

1.5 points for assignment (or equivalent number of statements)