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
Final Examination
Dec 13, 2017

Professor: Yongyi Mao, Mohamed Hossam Ahmed, Fadi Malek, Gilbert Arbez

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 60

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

Name: _____ **Student Number:** _____

Part 1:	25	
Part 2:	20	
Part 3:	15	
Total:	60	

⇐ Do not write in this box!

Number of pages: 10

Part 1 – Short Answer Questions

1) (4 points) Define a new type DATA_POINTS which contains the following members:
   - *identifier* (a string of up to MAX_STR_SIZE characters),
   - *numPoints* (an integer that gives the number of valid data points),
   - *points* (a 2D array that contains a number of real value x, y point coordinates), MAX_POINT_SIZE is a symbolic constant that gives the number of points that can be stored in the 2D array and assume that these points shall be used for plotting purposes).

C Code

2) (3 points) 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).

C Code

3) (3 points) Provide the output of the following C code in the adjacent box?

```
void fixup(int nuts, int *fruit);
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

4) (4 points) Show what the following program prints in the adjacent box.

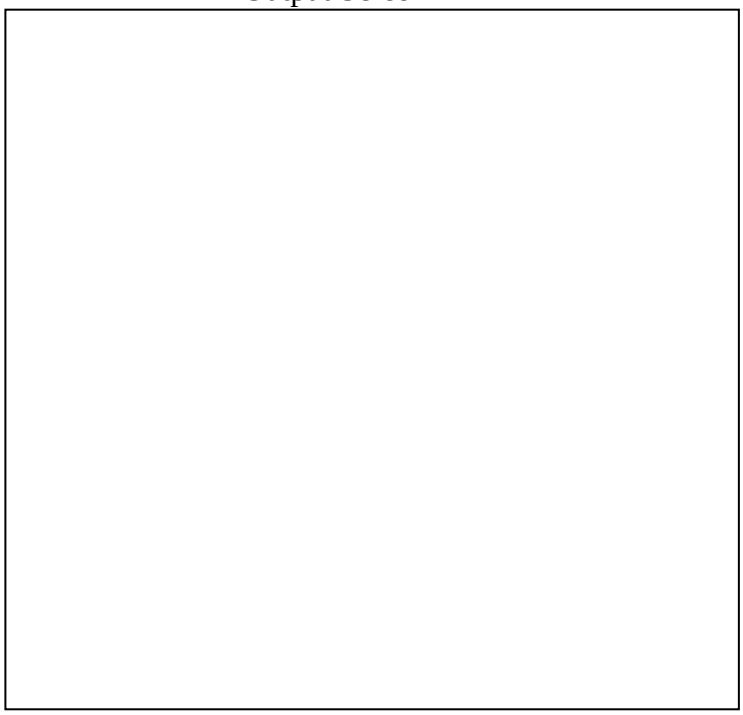
```
#include <stdio.h>
#define N 3

// a structure type for arrays.
typedef struct
{
    int *ptrData;
    int length;
} ARRAY;

void printArray(ARRAY a)
{
    int i;
    printf("Array Content:\n");
    for (i=0; i<a.length; i++)
    {
        printf("%d\n", a.ptrData[i]);
    }
}

int main()
{
    int x[N]={1, 3, 5};
    ARRAY array1;
    ARRAY array2;
    array1.ptrData=x;
    array1.length=N;
    array2=array1;
    array1.ptrData[0]=10;
    printArray(array1);
    printArray(array2);
    return 0;
}
```

Output Screen



5) (2 points) Give the function call used to write to the open binary file, referenced by `fPtr` (of type `FILE *`), the contents of the 1-D array, `values`, of type `INPUT` that contains 20 elements.

6) (4 points) Complete the function `fileExists` which returns TRUE if the file exists and FALSE otherwise (assume that the symbolic constants TRUE and FALSE are defined). Ensure that NO file is left open and that the content of the file is NOT affected.

C Code

```
int fileExists(char *filename)
{
```

7) (5 points) Show what the following program prints in the adjacent box.

```
#include <stdio.h>

void mazeOfRecursion(int x)
{
    if (x==0)
        printf("dead end\n");
    else
    {
        if (x%3==0)
        {
            printf("turning right\n");
            mazeOfRecursion(x-1);
        }
        printf("turning left\n");
        mazeOfRecursion(x-1);
    }
}

int main()
{
    mazeOfRecursion(5);
    return 0;
}
```

Output Screen

Part 2 – Programming Exercises: computing algorithms (20 Points).

Question 1 (10 points): Searching Algorithm

Complete the following function that searches and counts the number of even numbers in an array.

```
/*-------------------------------------------------------------------------
Function: findNumberOfEvenNumbers
Parameters:
    n - number of elements in the array (its length)
   ptr - the address of the memory where the array of interest is stored
Return:  The number of even numbers in the array.
Description:  Finds the number of even numbers.
-------------------------------------------------------------------------*/
int findNumberOfEvenNumbers(int n, int *ptr)
{
```

Question 2 (10 points): Bubble Sort

The `CUBE` structure contains the color and dimensions of a cube. Complete the function `bubbleSortCubes` that sorts an array of `CUBE` structure values according to increasing volumes using the bubble sort algorithm. Call the function `getVolume()` to determine a cube's volume. Make your sorting function as efficient as possible.

```
#include <stdio.h>


typedef struct
{
    char color[20];  // cube color
    double w;  // width
    double h;  // height
    double d;  // depth
} CUBE;



#define TRUE 1
#define FALSE 0




// function prototypes
double getVolume(CUBE *);
void bubbleSortCubes(int, CUBE []);




/*-------------------------------------------------------------------------
Function: getVolume
Parameters:
    cPtr - reference cube structure variable
Description: Computes the volume of the referenced cube structure.
-------------------------------------------------------------------------*/
double getVolume(CUBE *cPtr)
{
    double volume;
    volume = cPtr->w * cPtr->h * cPtr->d;
    return(volume);
}
```

```
/*-------------------------------------------------------------------------
Function: bubbleSortCubes
Parameters:
    n - number of CUBEs in the array
    cubes - reference to array of CUBEs to be sorted
Description: Sorts the elements of the array according to the computed
             volumes of the cubes. The function is efficient.
-------------------------------------------------------------------------*/
void bubbleSortCubes(int n, CUBE cubes[]);
{
```

Part 3 – Numerical Methods (15 Points).

Question 1 (5 points): Numerical Equations

(a) (2.5 points) **Euler`s Method**: For the following differential equation

$$\frac{dy}{dx} = \tan(x/3)$$

- Give the difference question for solving *y(x)* using Euler's method, i.e. how to calculate values of y_{i+1} from the values of y_i and x_i.

- Assume that the values x_i and y_i are stored in the variables `xi` and `yi`, the variable `deltax` gives the step for x, provide a single C instruction that will calculate the values of y_{i+1} and store it in the variable `yi`.

(b) (2.5 points) **Trapezoidal Rule**

$$I = \int_1^2 \left(2x + \frac{3}{x}\right)^2 dx$$

- Apply the Trapezoidal rule to express the above definite integral numerically. That is, express *I* in terms of the step size *h* and a set of values of *x* ($x_0, x_1, \ldots, x_n$). Make your expression as simple as possible.

- Given 10 numerical steps, what will be the value of the step size h.

Question 2: Integration – The Trapezoidal Rule

The amount of heat (calories), H, required to heat material of mass *m* from temperature T_1 to temperature T_2 is given by the definite integral

$$H = m\int_{T_1}^{T_2} c(T)dT$$

where the heat capacity *c(T)* for the material is given by

$$c(T) = 0.132 + 1.56\times10^{-4}T + 2.64\times10^{-7}T^2$$

where T is temperature in ^oC and *c* has units of cal/(g ^oC).

Complete the following function that will compute the amount of heat, H, required to heat the material of mass *m*. Guidelines:

- Arrays are not required or allowed in your solution.
- Use a separate function to compute *c(T)*

```
#define N 10  // Number of integration steps
/*-------------------------------------------------------------
Function: calculateH
Parameters:
    temp1 - initial temperature of the material T1
    temp2 - final temperature of the material T2
    mass – the material's mass
Return Value
    The heat required to raise the material's temperature from T1 to T2.
Description: Computes the heat required to heat the material of mass m
             from T1 to T2 using the trapezoidal rule.
-------------------------------------------------------------*/
double calculateH(double temp1, double temp2, double mass)
{
```

(Part 3 Question 2 continued)