UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

APS 105 — Computer Fundamentals
Final Examination
April 14, 2016
Special Accommodations - Paper exam
(150 minutes)

Examiner: B. Korst, P.Eng.

For this Final Exam, you will use the programming techniques presented in the lectures, used in quizzes and labs, and covered in Chapters 1 through – and including – 11 of the ZyBook.

You can use your ZyBook, other books and notes or the Internet in any device as a reference, but you cannot exchange messages with anyone *in any way*. In addition, *programs copied from another student or straight, word for word, from the internet will result in a zero for the exam and an academic offence process.*

All questions are to be answered on the examination paper. If the space provided for a question is insufficient, you may use the extra pages to complete your answer. If you use the extra pages, please direct the marker to the page and indicate clearly on that page which question(s) you are answering there.

You must use the C programming language to answer programming questions.

Full Name: _____

Student Number: _____ UTORID: _____

MARKS

1	2	3	4	5	6	7	8	9	Total
/5	/5	/5	/10	/10	/15	/15	/15	/20	/100

Question 1 [5 Marks]

The code below was intended to create a recursive function to calculate a factorial, but it presents a stack overflow. Resolve the problems in the code.

```
/*
 * File:   Question01.c
 * A program with a significant problem
 */
#include<stdio.h>

int someOp(int n)
{
    if (n==0)
    {
        return 1;
    }
    return someOp(n)*n;
}

int main(void)
{
    int a, n;
    printf("Input an integer number between 1 and 10: ");
    scanf("%d", &n);

    a = someOp(n);

    printf("The result is: %d \n\n", a);

    return 0;
}
```

Question 2 [5 Marks]

Write two functions: one that allocates memory space for a type double and returns a pointer to the type doublePointer (defined for you), then a second function to ensure no memory is wasted when the space is no longer needed. Your main function will

(a) declare two pointers, `pi` and `e` of type `doublePointer`;

(b) call one of your functions to allocate space and return a pointer to the allocated space;

(c) assign the value 3.1415 to the location pointed at by `pi` and 2.718 to the location pointed at by `e`; and

(d) will release the memory space allocated in item b.

```
#include<stdio.h>
#include<stdlib.h>
typedef double *doublePointer;

doublePointer newDouble (void)
{
  //WRITE YOUR CODE HERE




}

void noLeak(doublePointer z)
{
 //WRITE YOUR CODE HERE

}

int main(void)
{
// write parts a), b) and c) here




    printf("%lf %lf \n\n", *pi, *e);

// write part d) here



    printf("%lf %lf \n\n", *pi, *e);
    return 0;
}
```

This page has been left blank intentionally. You may use it for answers to any questions.

Question 3 [5 Marks]

Write a full program in C that takes two inputs by the user and calls a function to check if the first number input is exactly divisible by the second number. The function should return a `bool` type, so your program should include the appropriate library in C. Your code should be inserted in the skeleton code given below.

```
#include <stdio.h>

//YOUR FUNCTION GOES HERE









int main(void)
{

    int m, n;

    printf ("Enter two integer numbers:\n");
    scanf("%d %d", &m, &n);

    //ENTER YOUR CODE WITH THE FUNCTION CALL HERE.

```

```
    return (0);
}
```

Question 4 [10 Marks]

Complete the definition of the function `bestRally` whose header is shown below. The parameter `price` gives a list of the daily closing prices of a stock and the parameter `listLength` gives the length of the `price` array. A *rally* for a stock is defined as a period in which the closing price of the stock either increases or stays the same. The function should return the difference in the price of the stock in the rally with the largest increase in price. If the stock price never rises, the function should return zero.

```
double bestRally (double price[], int listLength)
{
   // your code goes here









}
```

Question 5 [10 Marks]

The skeleton code below shows that the program will make use of command line arguments. This particular program is intended to read one string as an argument from the command line and to check if it is in the array that has been given. If the string entered matches one of those in the array, the program prints it. If it does not, the program must print the message "Error" and exit. Complete the program:

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

int main (int argc, char *argv[])
{
     char *univ[] = {"UOFT","QUEENS","UWO","MCGILL"};

  // YOUR CODE GOES HERE









}
```

Question 6 [15 Marks]

The value of the mathematical constant e can be approximated by the formula:

$$e \approx \sum_{n=0}^{m} \frac{1}{n!} = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots$$

Write a complete C program that calculates the value of e to a precision of 10^{-6}. You must output (`printf`) the value of e calculated by your program, as well as the number of terms (that would be the m in the equation) in the summation when you reached the required precision. The skeleton code is found below.

```
#include<stdio.h>

int main(void)
{
    // your code goes here









    printf("The value of e is approximately %lf.\n", sum);
    printf("The number of terms it took is %d.\n\n", n);

    return 0;
}
```

Question 7 [15 Marks]

The program given below calls a recursive function `quickSort` to sort an array of integers. Your task is to resolve the problems and sort the array of integers given. Some problems are in the way the Quick Sort algorithm is implemented, but other problems appear at other parts of the program. There is a total of **five** problems. These are NOT syntax problems; they may be logic, function calls misplaced, wrong parameters, etc. Point out the problem and how you would solve them.

```
#include<stdio.h>

int a[10]={6,5,9,20,3,8,3,7,1,2};

void quickSort(int values[], int low, int high)
{
    int i;
    int left = low;
    int right = high;

    if (left >= right)
        return;

    int pivot = values[left];

    while(left < right)
    {
        while(values[right] >= pivot)
            right--;

        values[left] = values[right];

        while(values[left] <= pivot)
            left++;

        values[right] = values[left];
    }
    values[right]=pivot;

    quickSort(values,low, left-1);
    quickSort(values, left+1, high);

    printf("\n\nThe sorted sequence is: ");
    for (i=0; i<10; i++)
        printf("%d   ", a[i]);

}

int main(void)
{
    int j;
    printf("The original sequence is: ");
    for (j=0; j<10; j++)
        printf("%d   ", a[j]);

    quickSort(a, 1, 20);

    return 0;
}
```

This page has been left blank intentionally. You may use it for answers to any questions.

Question 8 [15 Marks]

You are to write a function that is part of a program to calculate the equivalent load of two loads in parallel. It is known that a parallel equivalent is expressed by the equation below, where `Za` and `Zb` are the loads connected in parallel.

$$Z_{parallel} = \frac{Za * Zb}{Za + Zb} = \frac{Product}{Sum}$$

Both `Za` and `Zb` are complex numbers, each with a real and an imaginary value, as in $Z = A + jB$, with A being the real part and B the imaginary part. Both `Za` and `Zb` are of type `COMPLEX`, defined by:

```
typedef struct{
    double real;
    double imaginary;
}COMPLEX;
```

The functions that perform the complex product and the complex sum are already implemented. They are: `prodImp` and `sumImp`. Their output is a type COMPLEX and their prototypes are:

```
COMPLEX prodImp (COMPLEX m, COMPLEX n);
COMPLEX sumImp (COMPLEX p, COMPLEX q);
```

Write a function called `parallel` that will calculate the resulting load by means of a complex division between the product and the sum as shown above.
The steps to effect the complex division can be illustrated as follows for a generic complex number Z. An itemized explanation is also shown below.

$$\frac{Z_{num}}{Z_{den}} = \frac{(Z_{num} * Z_{new})}{Z_{real}} = \frac{R_{real} + jR_{imaginary}}{Z_{real}} = \frac{R_{real}}{Z_{real}} + j\frac{R_{imaginary}}{Z_{real}}$$

(a) Create a new complex number which is a copy of the denominator, but with the changed sign on the imaginary value. For example:

$$Z_{den} = C + jD \qquad \text{then} \qquad Z_{dnew} = C - jD$$

(b) Multiply the numerator by Z_{dnew}. The product is also complex, called R, which has a real and an imaginary component;

(c) The product between the denominator and Z_{dnew} above will result in a purely real number, which is calculated as follows:

$$\text{If} \quad Z_{den} = C + jD \quad \text{then} \quad Z_{real} = C^2 + D^2$$

(d) Divide the real and imaginary parts of R (both are type `double`) by Z_{real} (also a type `double`) calculated in 3).

(e) Return the complex value.

The two inputs to function `parallel` will be the two loads, and the prototype is:

```
COMPLEX parallel (COMPLEX t, COMPLEX v);
```

This page has been left blank intentionally. You may use it for answers to any questions.

Question 9 [20 Marks]

PART A – The data type presented below defines a node of a linked list. You will need the information below for this question and the next. However, the answer to one question does *not* affect the answers to the other.

```
typedef struct student{
    char codename[20];
    char nationality[20];
    int age;
    struct student* next;
}STUDENT;
```

The head and the tail of the linked list have been declared as

```
STUDENT *ListHead, *ListTail;
```

Write a function `insertStudent` to insert a new node. Make sure that this function:

- allocates memory as needed;
- inserts the new node at the end of the list, unless it is the first node (check it!).

Answer:

This page has been left blank intentionally. You may use it for answers to any questions.

PART B – Assuming the database using the linked list on the previous question is filled with all information, write a function called `checkAge` to count how many students are registered for a given age bracket. The call to the function should be:

```
x = checkAge(ListHead, 19, 22);
```

Note: you do not need the answer to the previous question to solve this question.

Answer:

This page has been left blank intentionally. You may use it for answers to any questions.