UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

APS 105 — Computer Fundamentals
Midterm Examination
February 26, 2019
1:10 p.m. – 2:55 p.m.
(105 minutes)

Examiners: P. Anderson, M. Badr, B. Li, A. Poraria

Exam Type A: This is a "closed book" examination; no aids are permitted.

Calculator Type 4: No calculators or other electronic devices are allowed.

All questions are to be answered on the examination paper. If the space provided for a question is insufficient, you may use the last page to complete your answer. If you use the last page, please direct the marker to that 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. You are not required write `#include` directives in your solutions. The code provided to you may not have `#include` directives either. Except those excluded by specific questions, you may use functions from the `math` library as necessary.

The examination has 15 pages, including this one. If you use the extra page(s) note this on the original question page.

First Name: ______________________ Last Name: ______________________

Your Student Number: ______________________

Your Lab Section: (ID or day/time) ______________________

MARKS

1	2	3	4	5	6	7	8	9	10	11	12	Total
/4	/4	/4	/4	/4	/8	/10	/12	/12	/13	/13	/12	/100

Question 1 [4 Marks]

Write a single C statement — which contains exactly one terminating semi-colon (‘;’), and does not contain brace brackets (‘{’ or ‘}’) — that evaluates one answer x for the Quadratic equation. The formula is:

$$x = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \tag{1}$$

Assume that all variables (i.e., x, a, b, and c) have already been declared as doubles. You can assume all of the C math library functions are available. Write your solution in the box below.

```
x = (-1 * b + sqrt(b*b - 4*a*c)) / (2*a);
```

Question 2 [4 Marks]

Write a single C statement that declares a boolean-type variable named divisible and assigns true to divisible if and only if the value stored in the int variable named numOfItems is exactly divisible by 5 or 7. Assume that variable numOfItems has been declared and initialized. Write your solution in the box below.

```
bool divisible = numOfItems % 5 == 0 || numOfItems % 7 == 0;
```

Question 3 [4 Marks]

If i is declared as a variable of the int type, and p and q are both declared as variables to int-type pointers, which of the following assignment statements are illegal in the C programming language (**compile-time** warnings or errors), and which are legal (no warnings or errors at **compile-time**)?

Statements	Answer
`p = i;`	illegal
`*p = i;`	legal
`*q = &i;`	illegal
`p = q;`	legal

Question 4 [4 Marks]

If you try to compile the code below, it will generate errors/warnings. Identify the line (line numbers are shown on the left) where an error would occur, and write the code that would fix it.

```
1  #include <stdio.h>
2
3  void requestAge(int *age) {
4      do {
5          printf(”Please enter your age as a positive integer.\n”);
6          scanf(”%d”, &age);
7      } while(*age <= 0);
8  }
9
10 int main(void) {
11    printf(”How old are you?\n”);
12    int age;
13    requestAge(age);
14
15    printf(”Wow, you're %d years old!\n”, *age);
16    return 0
17 }
```

Solution:

Line #	*Corrected Line of Code*
6	scanf("%d", age);
13	requestAge(&age);
15	printf("Wow, you’re %d years old!\n", age);
16	return 0;

Question 5 [4 Marks]

Write the output of the following program.

Program	*Program Output*

```c
#include <stdio.h>

int main(void) {
  int *p, x;
  int fiveInt[5] = {1, 2, 3, 4, 5};
  int *q;

  p = NULL;
  q = fiveInt;
  x = 6;
  p = &x;

  printf("A: %d %d\n", x, *p);

  *(q+3) = *p;
  *p = *q + *(q+3);

  printf("B: %d %d %d\n", x, *p, *q);
  return 0;
}
```

Solution:

```
A: 6 6
B: 7 7 1
```

Question 6 [8 Marks]

Each of the following code segments may/will cause a **runtime error**. In the table, identify the potential runtime error and briefly explain how you would fix it.

Problematic Code	*What is the problem? How would you fix it?*

`char cArray[] = {'H', 'E', 'L', 'L', 'O'};`
`printf("The last character is %c.\n",`
`        cArray[5]);`
	Problem: The maximum index for cArray is 4 (array out-of-bounds error).
	Solution: Change [5] to [4].

`int i = 5;`
`while (i >= 0) {`
`  int j  = i * i;`
`  printf("j %d", j);`
`}`
`i--;`
	Problem: infinite loop (5 >= 0 will always be true).
	Solution: Decrement i inside the loop.

`int a, b, c;`
`printf("Enter two numbers.\n");`
`scanf("%d %d", &a, &b);`
 	
`c = a % b;`
`if (c > 3)`
`  printf("something\n");`
	Problem: b can be zero (divide by zero, floating point exception at runtime).
	Solution: Check if b == 0 first.

`#include <stdio.h>`
 	
`int higher(int *m, int *n) {`
`  int isHigher;`
 	
`  if (m >=n)`
`    isHigher = m;`
`  else`
`    isHigher = n;`
 	
`  return &isHigher;`
`}`
 	
`int main(void) {`
`  int c = 9, d = 8;`
`  int isHigher;`
 	
`  isHigher = higher(&c, &d);`
`  printf("%d\n", isHigher);`
 	
`  return 0;`
`}`
	Problem: In the function, pointer is assigned to a standard variable, Correction:
	`int higherCorrect(int *m, int *n) {`
	`  int isHigher;`
	`  if (*m >= *n)`
	`    isHigher = *m;`
	`  else`
	`    isHigher = *n;`
	`  return isHigher;`
	`}`

Question 7 [10 Marks]

Write a function called `median` to find the median among three integers, which is the number at the middle. Your function should take 3 `int`-type parameters and return the value of the number at the middle. For example, with integers 2, 7, and 5 as input, the function returns 5; with integers 6, 4, and 6 as input, the function returns 6.

Solution:

```
int median(int p, int q, int r){
   if((p >= q && p <= r) ||  (p >= r && p <= q))
      return p;
   else if ((q >= p && q <= r) || (q >= r && q <= p))
      return q;
    else
      return r;
}
```

Question 8 [12 Marks]

If you have a certain number of US dollars and wish to convert them to Canadian dollars, you could use the Canadian dollar to US dollar exchange rate (for example: 1 Canadian dollar = 0.75 US dollar). Write a complete C program that prompts its user for the current Canadian dollar to US dollar exchange rate (*e.g.* 0.75) and a value in US dollars, and then prints the value in Canadian dollars, **rounding to the nearest hundredth.** Your program will print the value with **6 digits after the decimal point.** Assume the user provides a valid exchange rate and US dollar amount.

Here is an example run of your program:

```
Enter the exchange rate (1 CAD = ? USD): 0.75
Enter the value in US dollars: 56
The value in Canadian dollars is 74.670000.
```

Solution:

```
int main (void) {
    double exchangeRate, cad, usd;

    printf("Enter the exchange rate (1 CAD = ? USD): ");
    scanf("%lf", &exchangeRate);

    printf("Enter the value in US dollars: ");
    scanf("%lf", &usd);

    cad = usd / exchangeRate;

    // rounding to the nearest hundredth
    double roundedCad = rint(cad * 100) / 100.0;

    printf("The value in Canadian dollars is %lf.\n", roundedCad);

    return 0;
}
```

Question 9 [12 Marks]

There are 0.3048 metres in a foot, 100 centimetres in a metre, and 12 inches in a foot. Write a program that will accept, as input, a length in feet and inches. **You do not have to check for valid input**—assume the user enters positive, non-fractional values for the feet and inches. The program will output the equivalent length in metres and centimetres (rounded to the nearest centimetre).

Your code should include four functions: one for input, one for output, one to perform the calculation, and main. The function prototypes are below. For full marks, your code should not use any global variables.

```
void getInput(int *outFeet, int *outInches);
void printOutput(int feet, int inches, int metres, int centimetres);
void convert(int feet, int inches, int *outMetres, int *outCentimetres);
```

An example of one run of the program is below:

```
Please enter the feet and inches to convert: 5 10
5 feet 10 inches is 1 metres and 78 centimetres.
```

Solution:

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

void getInput(int *outFeet, int *outInches) {
  printf("Please enter the feet and inches to convert: ");
  scanf("%d %d", outFeet, outInches);
}

void printOutput(int feet, int inches, int metres, int centimetres) {
 printf("%d feet %d inches is %d metres and %d centimetres.\n", feet, inches, metres, centimetres);
}

void convert(int feet, int inches, int *outMetres, int *outCentimetres) {
  double length = feet + (inches / 12.0);
  double metres = length * 0.3048;

  *outMetres = metres; // truncate to integer
  *outCentimetres = rint((metres - *outMetres) * 100);
}


int main(void) {
  int feet, inches;
  getInput(&feet, &inches);
  int metres, centimetres;
  convert(feet, inches, &metres, &centimetres);

  printOutput(feet, inches, metres, centimetres);

  return 0;
}
```

Question 10 [13 Marks]

Complete the definition of a C function `secondLargest` whose prototype is shown below. The function returns the index of the *second largest* integer in the `list` array, which contains `count` elements.

For example, if the list passed to the array is `{3, 9, 7, 5, 9, 8, 2, 4, 9}`, the function returns `5`, as `list[5]` contains the second largest integer `8`. If there are multiple occurrences of the second largest integer, the function returns the *first occurrence*. For example, if the list is `{3, 8, 3, 5, 9, 8, 2, 3, 8}`, the function returns `1`. If there does not exist a second largest integer (*i.e.*, all integers in the array are of the same value), the function returns `-1`. For the sake of simplicity, you may assume that all integers in the array are positive, and there exists at least one element in the array (*i.e.*, `count` > 0).

Solution:

```
int secondLargest(int list[], int count) {
    int largest = list[0], secondLargest = -1;
    int largestIndex = 0, secondLargestIndex = -1;

    for (int i = 1; i < count; i++) {
        if (list[i] > largest) {
            secondLargest = largest;
            secondLargestIndex = largestIndex;

            largest = list[i];
            largestIndex = i;
        } else if (list[i] < largest && list[i] > secondLargest) {
            secondLargest = list[i];
            secondLargestIndex = i;
        }
    }
    return secondLargestIndex;
}
```

Question 11 [13 Marks]

Write a function, `removeNegatives`, that removes negative values from an integer array. The function has two parameters: the array and its size. The function will modify the array by removing any negative values it has. The function will also return the number of negatives that it has removed.

When a negative value in the array is encountered, the remaining integers should be moved forward to fill in the gap and the negative value should be changed to zero. An example main function, and its output for three possible arrays, is provided below.

Example main Function:

```
int main(void) {
  const int SIZE = 8;

  // Potential array inputs.
  int allNegatives[] = {-3, -1, -5, -9, -10, -6, -7, -3};
  int adjacentNegatives[] = {-3, -1, 5, -9, -10, 6, -7, -3};
  int alternatingNegatives[] = {3, -1, 5, -9, 10, -6, 7, -3};

  // Pick an input.
  int *array = alternatingNegatives;

  int numRemoved = removeNegatives(array, SIZE);
  printf("%d negatives removed.\n", numRemoved);

  for(int i = 0; i < SIZE; i++) {
    printf("array[%d]: %d\n", i, array[i]);
  }
}
```

Outputs of Example main Function:

allNegatives	adjacentNegatives	alternatingNegatives
`8 negatives removed.`	`6 negatives removed.`	`4 negatives removed.`
`array[0]: 0`	`array[0]: 5`	`array[0]: 3`
`array[1]: 0`	`array[1]: 6`	`array[1]: 5`
`array[2]: 0`	`array[2]: 0`	`array[2]: 10`
`array[3]: 0`	`array[3]: 0`	`array[3]: 7`
`array[4]: 0`	`array[4]: 0`	`array[4]: 0`
`array[5]: 0`	`array[5]: 0`	`array[5]: 0`
`array[6]: 0`	`array[6]: 0`	`array[6]: 0`
`array[7]: 0`	`array[7]: 0`	`array[7]: 0`

Solution:

```
int removeNegatives(int array[], int size) {
  int count = 0;

  for(int i = 0; i < size; i++) {
    if(array[i] < 0) {
      count++;

      // Shift later values of the array forward.
      for(int j = i; j < size - 1; j++) {
        array[j] = array[j + 1];
      }

      // Index i was over-written, re-test it.
      i--;
      // "Delete" the negative.
      array[size - 1] = 0;
      // Shrink the array.
      size--;
    }
  }

  return count;
}
```

Question 12 [12 Marks]

In a Pascal's Triangle, the first row, row #0, has a single element 1. Each succeeding row elements are the sum of the two elements just above (if there is only one number just above, then that number is duplicated). So the first 5 rows (numbering from zero) are:

```
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

Looking at the last row, row #4, we have sums: $0 + 1, 1 + 3, 3 + 3, 3 + 1, 1 + 0$ (getting the values from the row above) to give us $1, 4, 6, 4, 1$. If we push this all left we get:

```
1
1  1
1  2  1
1  3  3  1
1  4  6  4  1
```

Write a function `calculatePascalRowSeven`, with the prototype given below, that calculates row #7 (the eighth row) of Pascal's triangle, iterating from row #0. Do an in-place calculation, so that the result ends up in `pascalRow[]`. Do not use any other array. The given `main()` function prints the result.

```c
void calculatePascalRowSeven(int pArray[]); //function prototype

int main(void) {

  // row #n has n + 1 elements
  int pascalRow[7 + 1] = {1, 0, 0, 0, 0, 0, 0, 0};

  calculatePascalRowSeven(pascalRow);

  printf("Row 7 is:\n");
  for (int i = 0; i <= 7; i++) {
    printf("%d ", pascalRow[i]);
  }

  printf("\n");
}

// Implement your function on the next page
```

```
void calculatePascalRowSeven(int pArray[]) {
```

Solution:

```
void calculatePascalRowSeven(int pArray[]) {
  for (int row = 1; row < 8; row++) {
    int oneToLeft = 0;

    for(int element = 0; element < 8; element++) {
        int saveElement = pArray[element];
        pArray[element] = oneToLeft + pArray[element];
        oneToLeft = saveElement;
    }
  }
}
```

This page has been left blank intentionally. You may use it for answers to any question in this examination.