UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE AND ENGINEERING

APS 105 — Computer Fundamentals
Midterm Examination
October 14, 2015
6:45 p.m. – 8:30 p.m.
(105 minutes)

Examiners: J. Anderson, B. Li, J. Rose

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. Except those excluded by specific questions, you may use functions from the `math` library as necessary.

The examination has 15 pages, including this one.

Circle your lab section — (**1 mark will be deducted** if you fail to indicate your lab section):

PRA0101	**PRA0102**	**PRA0103**	**PRA0104**
Monday	Thursday	Thursday	Thursday
4-6pm	3-5pm	12-2pm	10am-12pm

First Name: _____ Last Name: _____

Student Number: _____

MARKS

1	2	3	4	5	6	7	8	9	10	11	12	13	14	Total
/2	/2	/2	/2	/2	/2	/3	/3	/2	/6	/6	/6	/6	/6	/50

Question 1 [2 Marks]

In the following C program, how many times will the letter `D` be printed?

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

int main(void) {
  int i = 300;

  while (i > 0) {
    if (i % 2 == 0)
      printf("D");
    i--;
  }

  return (EXIT_SUCCESS);
}
```

Solution:

`150 times.`

Question 2 [2 Marks]

Given that `averageSpeed` is a `double` type variable that contains a value representing a car's average speed in metres/second for the past 30 minutes, write a single C statement that declares and initializes a `double` type variable called `distance`. The variable `distance` should be initialized to represent the distance travelled by the car in kilometres for the past 30 minutes.

Solution:

```
double distance = averageSpeed * 30 * 60 / 1000.0;
```

Question 3 [2 Marks]

Write a single C statement that declares an `int` variable `randomGuess`, and initializes it with a randomly generated *odd* number between `LOWER_BOUND` and `UPPER_BOUND` (inclusive). Assume that `LOWER_BOUND` and `UPPER_BOUND` are odd numbers declared as integer constants, and `UPPER_BOUND` is the larger value of the two.

Solution:

```
int randomGuess = rand() % ((UPPER_BOUND - LOWER_BOUND) / 2 + 1)
                  * 2 + LOWER_BOUND;
```

Question 4 [2 Marks]

In a C program, three `bool`-type variables have been declared, with the names `red`, `green` and `blue`, and are initialized to either `true` or `false`. Write a single C statement that declares and initializes a `bool`-type variable called `yellow` that is set to `true` when both the `red` and `green` variables are `true`, but the `blue` variable is `false`.

Solution:

```
bool yellow = red && green && !blue;
```

Question 5 [2 Marks]

The variable `numApples` is an `int` type variable representing the number of apples in a barrel. The owner of the apples is deciding whether to sell them in packages of 3 or 5 apples. Write a single C statement that declares and initializes an `int` type variable called `leftover`. `leftover` should be initialized to the minimum of two quantities: 1) the number of apples left over when the barrel of apples is packaged into groups of 3; 2) the number of apples left over when the barrel of apples is packaged into groups of 5. **Hint:** Use a function in the `math` library.

Solution:

```
int leftover = fmin(numApples % 3, numApples % 5);
```

Question 6 [2 Marks]

The following C program is supposed to calculate and print (to two decimal points accuracy) the square root of every second integer number from 1 to 101 inclusive beginning with 1. There are four lines that contain errors in this program, which are either syntax or logical errors.

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main(void) {
5    double root;
6
7    for (int number  = 1; number < 101; number += 2) {
8      number = sqrt((double) root);
9
10     printf("The Square Root of %0.2f is %.1f\n", number, root);
11   }
12   return (EXIT_DENIED);
13 }
```

For each line that has error(s), state, in the following table, the line number that error(s), and give the correct statement.

Solution:

Line Number	Corrected Line
7	`for (number = 1.0; number <= 101.0; number += 2.0) {`
8	`root = sqrt(number);`
10	`printf("The Square Root of %d is %.2f\n", number, root);`
12	`return (EXIT_SUCCESS);`

Note that some correctly identified that there should have been a `#include <stdlib.h>` for EXIT_SUCCESS to be defined.

Question 7 [3 Marks]

Print the output produced by an execution of the following C program.

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

int main(void) {
  char firstChar = 'A';
  char secondChar = 'U';

  printf("%c", firstChar + 3);
  firstChar += 4;
  printf("%c", firstChar);
  printf("%c", firstChar - 3);
  printf("%c", secondChar);
  firstChar++;
  firstChar++;
  printf("%c\n", firstChar);

  return (EXIT_SUCCESS);
}
```

Solution:

```
DEBUG
```

Question 8 [3 Marks]

What does the following program print?

```
#include <stdio.h>

int main(void) {
  int i, j, k;

  for (i = 10; i <= 14; i = i + 2) {
    for (j = 5; j > 0; j--) {
      k = i + j;
      printf("%d ", k);
    }
  }

  return (EXIT_SUCCESS);
}
```

Solution:

```
15 14 13 12 11 17 16 15 14 13 19 18 17 16 15
```

Question 9 [2 Marks]

There have been four plenary lectures so far in this course, and there is one question from each below. You should give a written answer for **two of these four questions.** The answer should be one or two sentences. If you answer more than two questions, you must indicate which two you wish to be graded; if you do not indicate which, then the first two in order will be graded.

Plenary Lecture 1 — Creative Application for Mobile Devices (Professor Jonathan Rose)
This lecture discussed the many capabilities of a smartphone, and many different applications of them in research and that have been deployed. It finished with two key messages relating to 1) What is possible with smartphones, and 2) the potential impact they can have. Give, in one sentence, the essence of those two key messages you were asked to 'take home.'

Solution:
1. Building applications is a very creative process.

2. If you can put it in a phone, it "scales" (can be deployed) to everyone at low cost.

Plenary Lecture 2 — Serial Software Entrepreneur (Bobby John)
What was the one question that Mr. John suggested that *every* company should ask itself in relation to software?

Solution: Every company should ask itself how can software be used to make what the company does more efficient or effective? Perhaps how to automate what it does.

Plenary Lecture 3 — Machine Learning and Big Data (Dr. Inmar Givoni)
Dr. Givoni gave a high-level description of how an artificial neural network learns something — for example they can learn what a human face looks like. Describe, in just a few sentences, that method.

Solution: The network is "trained" by showing it many pictures that have faces, and other pictures that do not have faces. The training process adjusts the neural network so that it correctly identifies the ones that are faces, and correctly identify those that are not.

Plenary Lecture 4 — From Software to Circuits (Professor Jason Anderson)
What are the advantages of implementing a computation in a hardware circuit compared to implementing it in software running on a computer?

Solution: Fast, more energy efficient, possibly cheaper/smaller. (Just need 2 of these three to get full marks).

Question 10 [6 Marks]

Write a complete C program that first prompts the user for a character. The program then determines if the character is an upper-case letter (A-Z), a lower-case letter (a-z), a digit (0-9), or a special character (all other cases). Recall that the upper-case letters have consecutive encodings in the ASCII table. Likewise, the lower-case letters have consecutive encodings in the ASCII table, and the digits do as well.

Here are four sample executions of the program that illustrate the four cases your program must handle:

```
Enter a character: A
UPPER CASE.

Enter a character: 1
DIGIT.

Enter a character: e
LOWER CASE.

Enter a character: !
SPECIAL.
```

Solution:

```
int main(void) {
  char myChar;
  printf("Enter a character: ");
  scanf("%c", &myChar);

  if ((myChar >= 'A') && (myChar <= 'Z'))
    printf("UPPER CASE.\n");
  else if ((myChar >= 'a') && (myChar <= 'z'))
    printf("LOWER CASE.\n");
  else if ((myChar >= '0') && (myChar <= '9'))
    printf("DIGIT.\n");
  else
    printf("SPECIAL.\n");

  return 0;
}
```

Question 11 [6 Marks]

Write a C function `reverseDigits`, the prototype of which is given below, that takes an integer argument `value`, and returns an integer with its digits reversed. For example, if the argument is `1234`, the function will return `4321`.

Solution:

```
int reverseDigits(int value)
{
    int sum = 0, lastDigit;

    while (value != 0)
    {
        lastDigit = value % 10;
        sum = sum * 10 + lastDigit;
        value = value / 10;
    }

    return sum;
}
```

Question 12 [6 Marks]

In 1706, John Machin, then a professor in London UK, devised a formula for calculating π:

$$\frac{\pi}{4} = 4 \cdot \arctan(\frac{1}{5}) - \arctan(\frac{1}{239})$$

where the $\arctan$ values can be calculated using the following formula:

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} \ldots - \frac{x^{299}}{299} + \frac{x^{301}}{301}$$

Write a C program that calculates and prints π using this method, printing 10 digits after the decimal point. You must define your own function called `arctan` to compute the $\arctan$ values, and you are not allowed to use the `atan` function from the `math` library. When your program is run, its output should be:

```
Pi = 3.1415926536
```

Solution:

```
double arctan(double x) {
  int sign = 1;
  double sum = 0;
  for (int i = 1; i <= 301; i += 2) {
    sum += sign * pow(x, i) / i;
    if (sign == 1)
        sign = -1;
    else
        sign = 1;
    // or sign = -sign;
  }
  return sum;
}

int main(void)
{
  printf("Pi = %.10lf\n", (4 * arctan(1 / 5.0) - arctan(1 / 239.0)) * 4);
  return 0;
}
```

Question 13 [6 Marks]

Write a complete C program that reads as input from the terminal, an `int` type variable called `index`, which will be in the range from 0 to 9 (your program may assume this). The program should then print a 10-line pattern, where each line contains all of the digits. The first line will begin with `index` and print all digits on the same line in ascending order until 9 is reached, at which point it will "wrap around" to 0 and print the remaining unprinted digits ascending until `index - 1`. The next line will repeat the pattern, beginning with `index + 1`.

Here are two sample executions of the program:

```
Enter index: 0
0123456789
1234567890
2345678901
3456789012
4567890123
5678901234
6789012345
7890123456
8901234567
9012345678

Enter index: 3
3456789012
4567890123
5678901234
6789012345
7890123456
8901234567
9012345678
0123456789
1234567890
2345678901
```

Solution:

```
int main(void) {
  int i,j;

  int index;
  printf("Enter index: ");
  scanf("%d", &index);

  for (i = 0; i < 10; i++) {
    for (j = 0; j < 10; j++) {
      printf("%c", '0' + (index + i + j) % 10);
    }
    printf("\n");
  }
  return 0;
}
```

Question 14 [6 Marks]

Write a complete C program that prompts the user to input a single character repeatedly. When the *most recent three* characters form the pattern `NAN` the program should print out, on a separate line, `Pattern NAN found!` The program should finish and print the word `Done` on a separate line when the character `'F'` is entered.

Here is a sample execution of the program that illustrates how it operates:

```
Input Next Character: T
Input Next Character: N
Input Next Character: B
Input Next Character: N
Input Next Character: A
Input Next Character: N
Pattern NAN found!
Input Next Character: A
Input Next Character: N
Pattern NAN found!
Input Next Character: N
Input Next Character: A
Input Next Character: A
Input Next Character: N
Input Next Character: A
Input Next Character: N
Pattern NAN found!
Input Next Character: B
Input Next Character: A
Input Next Character: N
Input Next Character: A
Input Next Character: N
Pattern NAN found!
Input Next Character: A
Input Next Character: N
Pattern NAN found!
Input Next Character: F
Done
```

Solution:

```
int main(void) {
    const char FINISH_CHAR = 'F';
    const char C1 = 'N', C2 = 'A', C3 = 'N';

    char inChar1 = 0, inChar2 = 0, inChar3 = 0, nextInput;

    do {
        printf("Input Next Character: ");
        scanf(" %c", &nextInput);

        if (nextInput != FINISH_CHAR) {
            inChar1 = inChar2;
            inChar2 = inChar3;
            inChar3 = nextInput;

            if (inChar1 == C1 && inChar2 == C2 && inChar3 == C3)
                printf("Pattern %c%c%c found!\n", C1, C2, C3);
        }

    } while (nextInput != FINISH_CHAR);

    printf("Done\n");

    return (EXIT_SUCCESS);
}
```

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