UNIVERSITY OF TORONTO

FACULTY OF APPLIED SCIENCE AND ENGINEERING

FINAL EXAMINATION, DECEMBER 2003

First Year – Engineering Science

CSC 180H1F – INTRODUCTION TO COMPUTER PROGRAMMING

Exam Type: C

Examiner – T.F. Fairgrieve

Aid sheet must be hand-written.

No calculators.

Student Number: _____

Last Name: _____

First Name: _____

This final examination consists of 8 questions on 12 pages (including this one). When you are told to start, please make sure that your copy of the examination is complete. Answer each question directly on the examination paper, or on the back of a page if necessary.

This exam is SINGLE-sided.

1: _____/ 8
2: _____/12
3: _____/ 5
4: _____/10
5: _____/10
6: _____/15
7: _____/15
8: _____/15
TOTAL: _____/90

Question 1. [8 MARKS]

Consider the following C program:

```
#include <stdio.h>
int mi( int k ) {
    printf( "%d,", k );
    return k/2;                    /* C */
}
int re( int j ) {
    printf( "%d,", j );
    return mi( j*4 );              /* B */
}
int fa( int i )
{
    int x = 0, k;
    printf( "%d,", i );
    for ( k = 1; k <= i; k += 2 ){
        x += mi( re( k*2 ) );      /* A */
    }
    return x;
}
int main()
{
    printf("%d\n", fa( 4 ));
    return 0;
}
```

Part (a) [4 MARKS]

Trace through this C program **carefully** and show what output is displayed when it is run.

Part (b) [2 MARKS]

If a represents the number of times that the statement labeled `A` is executed, and c represents the number of times that the statement labeled `C` is executed, describe the relationship between a and c.

Part (c) [2 MARKS]

If an `int` variable named `n` has a positive value, and the function call `fa( n )` is executed, how many times is the line labeled `B` executed? Circle the expression that best answers the question.

`1` `log(n)` `sqrt(n)` $\left\lceil \frac{n}{2} \right\rceil$ `n` n^2 n^3 `n log(n)`

Question 2. [12 MARKS]

Part (a) [4 MARKS]

Consider the following C program:

```
#include <stdio.h>
int a( int b ) {
    int result;

    printf("b = %d\n", b);
    if ( b > 2 ) {
        result = a(b-1) + a(b-2);
    } else {
        result = 1;
    }
    return result;
}
int main()
{
    printf("%d\n", a( 5 ));
    return 0;
}
```

Trace through this C program **carefully** and show what output is displayed when it is run.

Part (b) [4 MARKS]

If `i` is a variable and `p` points to `i`, which of the following expressions are aliases for `i`?

`(a) *p` `(c) *&p` `(e) *i` `(g) *&i`

`(b) &p` `(d) &*p` `(f) &i` `(h) &*i`

Circle all of the expressions that you think are aliases for `i`.

Part (c) [4 MARKS]

Consider the following C program:

```
#include <stdio.h>
#define N 25
int main()
{
    int i;
    char words[3][N] = {"x", "y", "z"};
    char w[N], *set[3];

    for ( i = 0; i < 3; i++ ) {
        set[i] = words[i];
    }

    scanf("%s", w);
    set[0] = w;

    scanf("%s", w);
    set[1] = w;

    for ( i = 0; i < 3; i++ ) {
        printf(" %s \n", set[i] );
    }

    return 0;
}
```

If this program were run with the input data

```
EngSci
is
easy
```

what output would be displayed?

Question 3. [5 MARKS]

Write an efficient C function named `check( x, y, n )` that returns 1 if both `x` and `y` fall between 0 and `n - 1`, inclusive. The function should return 0 otherwise. Assume that `x`, `y` and `n` are all of type `int`.
Write the function below.

```
#define TRUE   1
#define FALSE  0

/* Function to determine whether or not both x and y fall
   between 0 and n-1, inclusive.                          */
```

```
}
```

Question 4. [10 MARKS]

Write an efficient C function named `digit( n, k )` that returns the k^{th} digit (from the right) in the positive integer `n`. For example, `digit(829,1)` returns `9`, `digit(829,2)` returns `2` and `digit(829,3)` returns `8`. If either `k` or `n` is not positive, or if `k` is greater than the number of digits in `n`, the function should return `-1`.

Question 5. [10 MARKS]

Consider the problem of trying to accurately evaluate the mathematical function

$$f(x) = \ln(x + \sqrt{x^2 + 1}) \qquad (*)$$

using floating-point arithmetic.

Part (a) [2 MARKS]

For what values of x is it difficult to accurately evaluate expression $(*)$ using floating-point arithmetic? Justify your response.

Part (b) [4 MARKS]

Give an equivalent formula for expression $(*)$ that could be accurately evaluated using floating-point arithmetic for the values identified in part (a). Briefly justify your response.

Part (c) [4 MARKS]

Write a C function that could be used to accurately evaluate $f(x)$. You may make use of the math library functions `sqrt()` (for evaluating $\sqrt{}$) and `log()` (for evaluating ln()) in your solution.

Question 6. [15 MARKS]

Suppose we have a one-dimensional array `A` of `n` integers that we wish to sort into nondecreasing order. One way to accomplish this task is to apply an algorithm called Selection Sort. In this algorithm we find the largest element in a given list and swap it with the element at the end of the list. We then find the second largest element and swap it with the element in the second-to-last position. And so on.

Part (a) [10 MARKS]

Write a **recursive** C function that uses the Selection Sort algorithm to sort a one-dimensional array of `n` integers. (You **must** write a recursive function. A nonrecursive solution will earn no marks.)

Part (b) [5 MARKS]

Describe test cases that you would use to judge whether or not your `selectionSort` function works correctly. The reason for performing each test should be significantly different than the reasons for the other tests. (**You can answer this part even if you have not solved part (a).**)

	Description of list to be sorted	Reason for performing the test.
1		
2		
3		
4		
5		
6		

Question 7. [15 MARKS]

A **magic square** is an arrangement of the numbers from 1 to n^2 in an $n \times n$ array, with each number occurring exactly once, and such that the sum of the entries of any row, any column, and either diagonal are the same. It is not hard to show that this sum must be $\frac{n(n^2+1)}{2}$. The number n is called the **order** of the magic square.

The 3×3 array

$$\begin{bmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{bmatrix}$$

is a magic square because it satisfies the properties listed above.

Write an efficient and complete C program that:

1. prompts the user to enter an order n
2. reads an $n \times n$ array of integers
3. prints a message that indicates whether or not the input array is a magic square

You may assume that the user will enter a positive order n that is less than or equal to 10.

Place your Question 7 solution on the next page.

Please enter your Question 7 solution here.

Question 8. [15 MARKS]

For data compression and other purposes, we often need to look for runs (or sequences) of repeated characters in data. For example, in the string

```
abbbbccdeeeeef
```

there is a run of 4 `b`'s, a run of 2 `c`'s and a run of 5 `e`'s. We also consider that the `a`, the `d`, and the `f` are all runs of length 1.

Part (a) [13 MARKS]

Write an efficient C function named `compressString` that, given an input character string, builds a character string that contains a possibly compressed representation of the input character string. The new representation is determined by counting the lengths of runs of a single character, as demonstrated below:

- Given the string `"abbbbccdeeeeef"`, `compressString` builds the string `"1a4b2c1d5e1f"`.
- Given the string `"a"`, `compressString` builds the string `"1a"`.
- Given the string `"zzzzzzzzzzzzzzzzzzzzzzzzzz"`, `compressString` builds the string `"26z"`.
- Given the string `"xxbbbxx"`, `compressString` builds the string `"2x3b2x"`.
- Given the string `"aaAa"`, `compressString` builds the string `"2a1A1a"`. (Upper-case and lower-case letters are different.)
- Given the string `""`, `compressString` builds the string `""`.

Write your `compressString` function on the next page. State any assumptions that you make. The original string should not be changed. Note that the function `sprintf` may be used to determine the character string representation of a numerical value stored in an `int` variable. For example, if the variable `s` is an array of characters and `int i = 42;`, the function call `sprintf( s, "%d", i )` sets `s` to the string `"42"`. You may use the library functions `strlen()` and `strcpy()` in your solution.

Please enter your Question 8 (a) solution on next page.

Part (b) [2 MARKS]

Give an example of a **compressed** string built by the above algorithm that cannot be unambiguously (or uniquely) **un**compressed. Explain.

Please enter your Question 8 (a) solution here.

Total Marks = 90