**University of Toronto**
**Faculty of Applied Science and Engineering**

**Final Examination, December 2001**

**CSC 180H1F: Introduction to Computer Programming**

**Duration: 2½ hours**

**Exam Type: D**
**Aids allowed: One 8½x11" aid sheet (both sides). No calculators.**

**Examiner: Alan Rosenthal**

Last name:

First name(s):

Student number:

*Make sure you have all seven pages (including this page).*

**Note:** Pay attention to whether the question asks for a complete program (with #include, for example, although a prologue comment is not required for this exam) or asks only for a function.

**Note:** Answers not in the correct space will not be graded unless a note in the correct space says "see page ..." and the answer on that page is clearly labelled with the question number.

Be careful not to get stuck on some questions to the complete exclusion of others.
The amount of marks allotted does not necessarily indicate how long it will take you to complete the question, nor does the size of the answer-space indicate the size of the correct answer.

Leave this table blank.

| question | grade | of | question | grade | of |
|---|---|---|---|---|---|
| 1 | | 7 | 6 | | 10 |
| 2 | | 10 | 7 | | 9 |
| 3 | | 16 | 8 | | 15 |
| 4 | | 8 | 9 | | 15 |
| 5 | | 10 | subtotal | | |
| subtotal | | | total | | 100 |

1. [7 marks] (the "philosophical" question)

Here is a code fragment which calls printf. Printf is strange because it can take different numbers of arguments.

```
if (x == 0)
    printf("No glops were found.\n");
else
    printf("%d glops were found.\n", x);
```

The following code fragment calls a hypothetical void function "p":

```
if (x == 0)
    p(5);
else
    p(5, x);
```

Why is this problematic? That is, why is the "p" example an impossible situation even though the "printf" example is workable?

2. [10 marks] Write a program fragment (doesn't even need to be in a function) which computes the lowest number $n$ greater than 1000 such that $n = 3k^2$ for some integer $k$.

3. [16 marks] Consider the following declarations and initializations. The array initialization you see sets a[0] to be 12, a[1] to be 21, and a[2] to be 31.

```
int n = 13;
int m = 6;
double x = 10;
char s[8] = "abcde";
int a[3] = { 12, 21, 31 };
```

For each of the following expressions, give both the type and the value of the expression. (Use the most simplified form for each value; e.g. do not write "3+5" when you mean 8.) An example is given.
Note that the ASCII value of 'a' is 97, 'b' is 98, 'c' is 99, and so on.

| Expression | Type | Value |
|---|---|---|
| `a[0] + m` | `int` | `18` |
| `s[2]` | | |
| `n % m` | | |
| `n / m` | | |
| `a + 1` | | |
| `n / x` | | |
| `s[2] + x` | | |
| `x + a[1] / a[2]` | | |
| `n + m / x` | | |

4. [8 marks] What does this program fragment output? Again, the letters are in order in the ASCII character code; for example, `'a'+3` is `'d'`.

```
int i, j;
for (i = 0; i < 6; j++) {
    j = i * 3 + 1;
    if (j > 10)
        putchar('a' + (j - 10));
}
```

5. [10 marks] Write a recursive function which takes two parameters, a string and a character, and outputs the longest final string (a substring which goes up to the end of the string)† which starts with that character.

For example, final("hello", 'l') will return the string "llo" (a pointer to the first 'l', i.e. its argument plus two, by pointer arithmetic), and final("ababdef", 'b') will return the string "babdef".
Since this function's return type will be a pointer (char *), we will return NULL if the character does not actually appear in the string. So final("hello", 'g') will return NULL.

The recursive call can pass s+1 (where s is the string parameter) to pass in a string lacking the first character, using pointer arithmetic.
(This is only valid if s is not the empty string, i.e. if s[0] != '\0' — we may not look at the array past the '\0'.)

The function header is provided. The function must be written recursively (no explicit loops).

```
char *final(char *s, char c)
{
```

† For example, the complete set of final strings of "hello" is: "hello", "ello", "llo", "lo", "o", and "".

6. [10 marks] Recall the infinite series we discussed which sums to $\pi$:

$$\pi = \frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \ldots$$

Write a function named "pi" which takes an integer parameter which is the number of terms of this series to add together, and returns a double value which is the sum. For example, pi(1) returns 4. The function header is provided.

```
double pi(int n)
{
```

7. [9 marks] Here is a recursive factorial function:

```
int factorial(int n)
{
    if (n < 2)
        return 1;
    else
        return n * factorial(n - 1);
}
```

Here is a function which recurses infinitely if its parameter is 5, otherwise it is the identity function:

```
int funnyidentity(int n)
{
    if (n == 5)
        return funnyidentity(n);
    else
        return n;
}
```

Can you make a recursive function which recurses infinitely if its parameter is 5, otherwise is a recursive factorial function?

8. [15 marks] In this problem, everyone has one or zero best friends. People are represented by non-negative integers.

There is a "friend" library, with appropriate declarations in a system include file <friend.h>. Among other useful functions for manipulating friends, it contains a function "bestfriend" which takes one integer argument and returns that person's one best friend, or -1 if that person doesn't have a best friend.

a) What is the 'extern' declaration of bestfriend which will appear in the system include file friend.h?

b) There is an implicit chain of best friends. If 3's best friend is 5, and 5's best friend is 8, and 8's best friend is 6, then we could say that 6 is an "eventual best friend" of 3. Since everyone has only one best friend (or none), if you keep going down this chain, you eventually find all of the starting person's "eventual best friends". (Assume that there are no loops, e.g. if 3's best friend is 5, then 5's best friend is not going to be 3.)

Write a complete C program which uses the bestfriend function to determine whether or not 9 is 2's eventual best friend. (Your program takes no input.)

You do not write the "bestfriend" function; it is in the "friend" library.

9. [15 marks]
Write a complete program which repeatedly prompts the user for a file name, then attempts to open the file, and if it succeeds it reads and displays the first *character* of the file (read it with getc(fp)). If it fails, it must call perror(). It does this repeatedly in a loop until the user types "quit".

(Since this is in a loop, you must fclose() the open file before opening the next one (or your program will stop working after a set number of files which is the limit on how many files you can have open at once).)

Helpful snippet (use as you like below, or not (you can say "above stuff goes here" or copy it)):

```
if (fgets(buf, sizeof buf, stdin) == NULL)
    return 0;
if (strcmp(buf, "quit\n") == 0)
    return 0;
```

Extra space if needed (you *must* write "see page 8" in the usual answer space for the question)