CSC 180 midterm #2 solutions

21 November 2001

1. [7 marks] The Towers of Hanoi solution takes a number of moves which is a function of *n* and can be defined recursively:

- *nmoves*(1) = 1
- For $n > 1$, $nmoves(n) = 1 + 2 * nmoves(n - 1)$

Use this recursive definition of *nmoves* to write a C function "nmoves" which takes one int parameter and returns an int which is the number of moves in the solution for *n* disks. You can assume that the parameter $n \geq 1$.

Answer:

```
int nmoves(int n)
{
    if (n == 1)
        return 1;
    else
        return 1 + 2 * nmoves(n-1);
}
```

2. [5 marks] The following complete program gives the wrong answer for $\sqrt{2}$. "sqrt" is a function in the math library.

```
#include <stdio.h>

int main()
{
    double q = sqrt(2.0);
    printf("%g\n", q);
    return 0;
}
```

a) Why does it give the wrong answer?

Answer: It doesn't #include <math.h>, so sqrt is not declared.

b) If compiled with `gcc -Wall`, what error (warning) message would the compiler give? (Obviously, an exact answer is not required here; what you need to do is specify the approximate complaint it would make.)

Answer:

warning: implicit declaration for function sqrt

or something about the fact that sqrt is not declared. It will *not* advise you to #include <math.h>; it doesn't know where a declaration can be found, just that it's missing.

3. [10 marks] Write a function called “maxarray” which takes two parameters, indicating an array of ints and its size, and returns the maximum (largest) int in the array.

Answer:

```
int maxarray(int *a, int size)  /* assumes size >= 1 */
{
    int i;
    int max = a[0];

    for (i = 0; i < size; i++)
        if (a[i] > max)
            max = a[i];

    return 0;
}
```

4. [10 marks] The following function exchanges the first and last characters of a string of length 5 (not counting the \0). (Working only with strings of length 5 is a serious limitation, obviously.)

```
void reverse5part(char *s)
{
    char t = s[4];
    s[4] = s[0];
    s[0] = t;
}
```

Thus the following sequence:

```
char s[10];
strcpy(s, "squid");
reverse5part(s);
printf("%s\n", s);
```

will output “dquis”.

Write a function “reverse” which takes any length string and reverses it completely.
For example, reversing “squid” will yield “diuqs”, and reversing “supercalifragilisticexpialidocious” will yield “suoicodilaipxecitsiligarfilacrepus”.
Your function takes only one parameter, just like reverse5part above.

[hint about strlen removed from solutions to make this fit on a page]

Answer:

```
void reverse(char *s)
{
    int i;
    int len = strlen(s);
    int last = len - 1;  /* last char of string */
    int howfar = len / 2;  /* how far to reverse: only half-way! */

    for (i = 0; i < howfar; i++) {
        char t = s[last - i];
        s[last - i] = s[i];
        s[i] = t;
    }
}
```

5. [8 marks] Write a complete program which opens a file named ‘‘essay’’, reads all the lines of the file with fgets(), and displays the count of the number of lines in the file.

If the fopen() fails, you must call perror() to print an appropriate error message.

You can assume that no line in the file is longer than 123 characters (not including the \n at the end of the line).

The three parameters to fgets are, in order: space for the string (i.e. a pointer to the zeroth element of a char array), the size of the string space (including the \n and \0, i.e. the actual size of the array), and the FILE pointer.

Answer:

```
#include <stdio.h>

int main()
{
    int count;
    char buf[125];  /* line of 123 characters plus ’\n’ plus ’\0’ */

    FILE *fp = fopen("essay", "r");
    if (fp == NULL) {
        perror("essay");
        return 1;
    }

    for (count = 0; fgets(buf, sizeof buf, fp); count++)
        ;

    printf("%d\n", count);
    return 0;
}
```