Sample Final Answers

These are sample questions that are very similar to the ones I will ask on the midterm.

1. True or False: the value of `isdigit(9)` is 1.

   Answer: False. the character with value 9 is a control character (a horizontal tab).

2. True or False: *In general*, recursion is more efficient than iteration.

   Answer: False. Recursion may be more efficient in a few cases, but in general, iteration avoids the overhead of the function call.

3. Multiple choice: I want to copy a string stored in an array `src` to another array called `dest`. Which of the following statements will do this?

   (a) `dest = src;`

   (b) `dest[] = src[];`

   (c) `strcpy(dest, src);`

   (d) `while(*dest = *src);`

   (e) `strcmp(dest, src);`

   Answer: c. a copies a pointer, which is illegal as `dest` is declared as an array, and so is a constant; b is illegal as the use of `[]` in this context is a syntax error; d is an infinite loop, and e compares the strings.

4. Multiple choice: Which of the following describes a recursive function?

   (a) It is a function with no parameters.

   (b) It is a function that contains loops to count something.

   (c) It is a function that calls itself.

   (d) It is a function that returns a value.

   (e) It is a function that is called to end the program.

   Answer: c. The others may be true of any function, whether it is recursive or not.

5. Here is a macro for computing the maximum of two numbers:

   ```
   #define max(x, y) ((x) > (y) ? (x) : (y))
   ```

   and here is a function:

   ```
   int max(int a, int b)
   {
        return(a > b ? a : b);
   }
   ```

   In the following code fragment, give the values of `a`, `b`, `c`, `d`, and `e` at the end if `max` is the above macro, and then if `max` is a function call:

   ```
   a = -1;
   b = 0;
   c = 1;
   d = max(++a, b);
   e = max(c++, b);
   ```

variable	value if `max` is macro	value if `max` is a function
a		
b		
c		
d		
e		

Answer:

variable	value if `max` is macro	value if `max` is a function
a	0	0
b	0	0
c	3	2
d	0	0
e	2	1

6. Here is a very simple program which is designed to sum its arguments:

```
#include <stdio.h>

int main(int argc, char *argv[])
{
    int sum = 0;
    int i;

    for(i = 1; i < argc; i++)
        sum += argv[i];
    printf("%d\n", sum);

    return(0);
}
```

   (a) Suppose this program is invoked as

   ```
   sum 1 2 3
   ```

   What are the values of `argc` and the elements of `argv`? Draw a picture if that would help you indicate what the values are. Please note I want the value of `argc`, `argv`, and each element of `argv`, and so forth, either as a number or character, or as arrows in a picture (for addresses).

   (b) When I ran this programs, I got 2093422869. Why does the program print the incorrect value, and how would you fix it? Feel free to us any library functions you might find helpful. (If you cannot write the code, for partial credit simply say what you would do.)

Answer:

(a) Here, `argc` is 4. `argv` contains the address of `argv[0]`. The value of `argv[0]` is the address of the string `"sum"`, the value of `argv[1]` is the address of the string `"1"`, the value of `argv[2]` is the address of the string `"2"`, the value of `argv[3]` is the address of the string `"3"`, and the value of `argv[4]` is the **NULL** pointer.

(b) The program adds in the *addresses* of the strings, not the numbers those strings represent. The easiest way to fix the program is to replace the line

```
sum += argv[i];
```

with the line

```
sum += atoi(argv[i]);
```

or something similar.

7. Here is a function:

```
int mystery(char a[])
{
    register int i;

    for(i = 0; a[i]; i++)
        if (!isdigit(a[i]))
            return(0);
    return(1);
}
```

   (a) What does the above function do? Its purpose can be stated in a succinct sentence; for partial credit, you can describe what each line does, but for full credit, you must state its function in one very short sentence.

   (b) Rewrite the function using pointers rather than an array.

   Answer:

   (a) It returns 1 if the array `a[]` contains only digits. Otherwise, it returns 0.

   (b) Using pointers, this becomes:

```
int mystery(char *a)
{
    for( ; *a; a++)
        if (!isdigit(*a))
            return(0);
    return(1);
}
```

8. Rewrite the following into a Linux command or commands that do not use temporary files.

```
tr A-Z a-z < /usr/share/dict/words > /tmp/X
grep '^bananna$' /tmp/X > /tmp/Y
wc -l /tmp/Y
```

   Answer: Use pipes:

```
tr A-Z a-z < /usr/share/dict/words | grep '^bananna$' | wc -l
```

9. What is the value of `n` after the following code executes?

```
char charray[] = "A-DF123";
int n, s = 1;
char *p;

for(n = 0, p = charray; *p; p++){
    switch(*p){
    case '-':
        s = -s;
        break;
    case '0':
```

```
        case '1':
        case '2':
        case '3':
        case '4':
        case '5':
        case '6':
        case '7':
        case '8':
        case '9':
            n = n * 10 + (*p - '0');
            break;
        }
    }
    n = s * n;
```

Answer: `n = -123`