Name: Lab Section:

CprE 288 Fall 2012 – Homework 3
Due Thu. Sept. 13 in the class

Notes:

- Homework answers must be typed using a word editor. Hand in a hard copy in the class.
- Late homework is accepted within three days from the due date. E-mail the word file to your instructor. *Late penalty is 10% per day (counting from the end of your class on the due date).*

Building and Testing Instructions: In this exercise, you may want to test your programs in the C programming environment of a department Linux server. You may use any departmental Linux servers (e.g. linux-1.ece.iastate.edu, linux-2.ece.iastate.edu and so on) to build and run your programs using gcc (GNU C complier).

Remotely access one of those servers. See http://it.engineering.iastate.edu/remote-access/#Linux-Remote-Access for remote access instructions. See http://www.washington.edu/computing/unix/unixqr.html for commonly used UNIX commands.

You may create your program using your favorite editor and then copy it into your U: drive. Your home directory on those servers is your U: drive.

To build a program, assuming the name of the program file is myprogram.c:

```
gcc –o myprogram myprogram.c
```

To test run the program:

```
./myprogram
```

Question 1: Function and Pointer (10 pts)

The following function is intended to swap the values of two variables:

```
// swap the contents of x and y
void swap (int x, int y)
{
  int tmp;

  tmp = x;
  x = y;
  y = tmp;
}

// example of using swap()
void call_swap()
{
```

Name: Lab Section:

```
  int a, b;
  …
  swap (a, b);
  …
}
```

The function does not work as intended. Explain Why. Then revise the code, including both `swap` and `call_swap` functions. Test your code, and then cut and paste it in the space below.

C function is pass-by-value, that means swap() has local copies to x and y, and the changes of x and y will be local to swap(). Revise code as follows:

```
// swap the contents of x and y
void swap (int *x, int *y)
{
  int tmp;

  tmp = *x;
  *x = *y;
  *y = tmp;
}

// example of using swap()
void call_swap()
{
  int a, b;
  …
  swap (&a, &b);
  …
}
```

Question 2: Switch Statement (10 pts)

Complete the following function that counts the number of each vowel, **namely a, e, i, o, u and y and their upper case**, in a C string. **You have to use a switch statement**. The results for a, e, i, o, u should be stored in an array of five element in the same sequence; for example, element 0 stores the result for a, element 1 for e, and so on. See the comments for more information. The following is an example how the function is to be called.

```
char message[] = ”Welcome to CprE 288”;
int counter[5];
int count_vowels(int *, char *);
```

Name: Lab Section:

```
int main()
{
   count_vowels(counter, message);
   // after the call, counter[] == {0, 3, 0, 2, 0, 0}

   // YOUR TESTING CODE
}

// Count the number of vowel letters in a string.
// "str" is the pointer to the input string.
// Return the number of vowel letters in the array
// referenced by "result". The value of array elements
// may not necessarily be zero in the beginning.
int count_vowels(int result[], char *str)
{
   // YOUR CODE
}
```

Test your code, and the cut and paste your whole program here.

```
int count_vowels(int result[], char *str)
{
   char ch;

   // clear result
   for (int i = 0; i < 5; i++)
      result[i] = 0;

   // scan the string for vowels
   while ((ch = *str++) != '\0') {
      switch (ch) {
      case 'a':
         result[0]++;
         break;
      case 'e':
         result[1]++;
         break;
      case 'i':
         result[2]++;
         break;
      case 'o':
         result[3]++;
         break;
      case 'u':
         result[4]++;
```

Name: Lab Section:

```
            break;
        }
}
```

Question 3: For, While, Do-while Loops (20 pts)

Compete the following function that counts the number of positive (greater than zero) integers in an array, using three types of loops. See the comments for more details. The three versions should behave exactly the same.

```
// Count the number of positive integers in an array
// X: the input array.
// N: the number of elements in the array, N > 0
// return: the counted number.
int count_positive(int X[], int N)
```

a. [10 pts] Complete the function using a for loop.

```
int count = 0;

for (int i = 0; i < N; i++) {
   if (X[i] > 0)
      count++;
}

return count;
```

b. [5 pts] Complete the function using a while loop.

```
int count = 0;

int i = 0;
while (i < N) {
   if (X[i] > 0)
      count++;
   i++;
}

return count;
```

c. [5 pts] Complete the function using a do-while loop.

```
int count = 0;
```

Name: Lab Section:

```
int i = 0;
if (i < N) {
   do {
      if (X[i] > 0)
         count++;
      i++;
   } while (i < N);
}
```

Question 4: String Operations, Pointers, and C Library Functions (20 pts)

You may only use pointers, not array, in the following exercises.

a. [5] Complete the following function that returns the length of string s.

```
int strlen(char *s)
{
   int len = 0;
   while (*s++)
      len++;
   return len;
}
```

b. [5] Complete the following function that copies the string s2 to s1. Assume that s1 has sufficient storage.

```
int strcpy(char *s1, char *s2)
{
   while (*s1)
      *s1++ = *s2++;
}
```

c. [5] Complete the following function that compares two strings s1 and s2 lexicographically. Return 0 if the two strings are equal, -1 if s1 is less than s2, or 1 if s1 is greater than s2. The following are examples of lexicographical comparison results:

"a" > "b", "ab" > "abc", "ABC" < "abc",

"cpre 288" > "cpre 185", "abc" == "abc"

Name: Lab Section:

```
int strcmp(char *s1, char *s2)
{
   // scan the two arrays until a mismatch is found
   while (*s1 && *s2 && (*s1++ == *s2++))
      {}
   if (*s1 == '\0') {
      if (*s2 == '\0')
         return 0;
      else
         return -1;

   }
   else if (*s2 == '\0')
      return 1;
   else if (*s1 < *s2)
      return -1;
   else          // note: at this time, *s1 == *s2 cannot happen
      return 1;
}
```

Note: "while (*s1 && *s2 && (*s1++ == *s2++))" can be changed to "while (*s1 && (*s1++ == *s2++))". The two are equivalent.