CDA 3101 Assignment 5
Due in class on Friday, Oct. 8 by 11:45am

Assignments submitted after Oct.8, 11:45am and before Oct. 11, 11:45 will be considered one day late. No assignments will be accepted after Oct. 11, 11:45am.

Submit the programs (.s or .asm file) as attachments for all the problems below, via E-Learning Assignment Tool. You must use the E-Learning Assignment Tool to submit your code. You cannot submit your code using email.

Also submit a hardcopy of these programs in class with your name and “CDA Assignment 5” clearly written on it.

Test all your programs using PCSPIM because the TAs will use PCSPIM to run and test the program files you submit electronically. So, please make sure that the files you submit electronically can be successfully loaded and executed in PCSPIM. If your file does not load successfully in PCSPIM, you will immediately get a 0.

Make sure to follow the procedure call conventions (pass arguments in $a0-$a3, pass return values in $v0-$v1, saving appropriate registers on stack wherever required). Not following these conventions will result in significant loss of points.

Do not change the structure of the programs below in any way. Translate them word by word. If you change the structure of the program, you will lose a significant number of points.
1. (20 points) Translate the following code into MIPS assembly language code. **You can assume each character takes one byte.**

```c
main( ) {
    char *inputStr = malloc(20); /* dynamically allocate 20 bytes */
    gets(inputStr); /* gets a string from user, stores it in the inputStr */
    Encrypt(inputStr);
    printf("%s", inputStr); //prints the encrypted string content
}

/* Encrypt takes address of a string as an argument and it changes each
   Letters in the string by the third letter after it */
void Encrypt( char *str)
{
    int i, len;
    len = GetLength(str);
    for (i=0; i < len; i++) str[i] = str[i] + 3;
}

/* GetLength( ) takes address of a string. It computes and returns the length of
   this string. */
int GetLength(char *arr) {
    int i=0;
    while(arr[i] != '\0') i++; /* '\0' has an ascii value of 0 */
    return(i);
}
```
2. (30 points) Translate the following code into MIPS assembly language code. This
code asks the user for a string, reverses it and prints the reversed string. **You can assume that the string entered by the user is no more than 50 characters long.**

**Note that you have to submit a recursive solution for this problem; an iterative program will not fetch any points.**

```c
char  str[50];    /* you can assume each character takes one byte */
void main() {
    printf("Enter a string: ");
    scanf("%s", str);    /* reads a string from console and stores it in str */

    int count = 0;
    for(int i=0; str[i] != '\0'; i++)
        count++;

    reverse(0, count);

    printf("Reversed String: %s", str);  // prints the reversed string
}

void reverse(int start, int end) {
    if(start < end)
    {
        char temp = str[start];
        str[start] = str[end-1];
        str[end-1] = temp;
        reverse(start+1, end-1);
    } else {
        printf("Reversed String: %s", str);
    }
}
```