CS 3443 Computer systems

Assignment 04: Full marks 100

Due date: 10/16/2022 (11:59 PM CT) **End date: 10/23/2022 (11:59 PM CT)**

Please consider these points below you proceed with the assignment:

- **Follow the instructions provided on each question.**
- **Make sure to follow how the grade is distribution for each question. It is given in green text inside the square bracket.**
- **For assembly programming, just making the program run is not important. You need to follow the right concept.**
- **You cannot write your own MIPS program that satisfy the program description and/or output.**
- **Your MIPS program should be equivalent to the given C program, following the instructions given below the program.**

Following problems are focused on function in MIPS.

1. Write a complete running equivalent MIPS programs for the following C program: **[19 Marks]**

```c
/*
Program description: Program to calculate LCM of two positive numbers
*/

#include <stdio.h>

int LCM(int n1, int n2, int gcd);

int main(){
    int n1, n2, n2Modified;
    int gcd, lcm;

    printf("Enter the first positive integers: ");                  // [0.25 mark]
    scanf("%d", &n1);                                                // [0.25 mark]

    printf("Enter the second positive integers: ");                 // [0.25 mark]
    scanf("%d", &n2);                                                // [0.25 mark]

    gcd = n1;                                                        // [0.25 mark]
    n2Modified = n2;                                                 // [0.25 mark]

    while(gcd != n2Modified){                                        // [2 mark]

        if(gcd > n2Modified)                                         // [2 mark]
            gcd -= n2Modified;                                       // [0.5 mark]
        else                                                         // [1 mark]
            n2Modified -= gcd;                                       // [0.5 mark]
    }

    lcm = LCM(n1, n2, gcd);                                          // [2 mark]

    printf("The LCM of two numbers %d and %d is %d.", n1, n2, lcm);  // [0.25 mark]

    return 0;                                                        // [0.25 mark]
}

int LCM(int n1, int n2, int gcd){
    int lcm = (n1 * n2) / gcd;                                       // [4 mark]

    return lcm;                                                      // [1 mark]
}
```

Instructions:

- Correct passing of values to the LCM function. [2 mark]
- Use of *jal* and *jr* function to call function and return from the function respectively. [2 mark]

2. Write a complete running equivalent MIPS programs for the following C program: **[21 Marks]**

```c
/*
Program description: Program to display all prime numbers between two positive numbers
*/

#include <stdio.h>

// user-defined function to check prime number
int checkPrimeNumber(int n) {                                          // [1 mark]
    int j, flag = 1;

    for (j = 2; j <= n / 2; ++j) {                                     // [3 mark]

        if (n % j == 0) {                                              // [2 mark]
            flag = 0;                                                  // [0.25 mark]
            break;                                                     // [0.5 mark]
        }
    }

    return flag;                                                       // [1 mark]
}

int main() {

    int n1, n2, i, flag;

    printf("Please enter the first integer: ");                        // [0.25 mark]
    scanf("%d", &n1);                                                  // [0.25 mark]

    printf("Please enter the second integer: ");                       // [0.25 mark]
    scanf("%d", &n2);                                                  // [0.25 mark]


    // find the lower (i.e. n1) and upper (i.e. n2) range
    if (n1 > n2) {                                                     // [1 mark]
        n1 = n1 + n2;                                                  // [0.5 mark]
        n2 = n1 - n2;                                                  // [0.5 mark]
        n1 = n1 - n2;                                                  // [0.5 mark]
    }

    printf("Prime numbers between %d and %d are: ", n1, n2);          // [0.25 mark]

    // Check all the numbers between n1 and n2 for prime
    for (i = n1 + 1; i < n2; ++i) {                                    // [2 mark]

        // flag will be equal to 1 if i is prime
        flag = checkPrimeNumber(i);                                    // [2 mark]

        if (flag == 1) {                                               // [1 mark]
            printf("%d\t", i);                                         // [0.25 mark]
        }
    }

    return 0;                                                          // [0.25 mark]
}
```

Instructions:

- Correct passing of values to the *checkPrimeNumber* function. [2 mark]
- Use of *jal* and *jr* function to call function and return from the function respectively. [2 mark]

Following problems are focused on the use of stack during function calls in MIPS.

3. Write a complete running equivalent MIPS programs for the following C program: **[48 Marks]**

```
/*
Program description: Program to concatenate two strings and count vowels and consonents
*/

#include <stdio.h>

void countVowelConsotant(char str[100]){
    int i = 0;                                                       // [0.25 mark]
    int  vCount=0, cCount=0;                                         // [0.5 mark]

    printf("String with just alphabets: ");                          // [0.25 mark]
    puts(str);                                                       // [0.25 mark]

    //'\0' signifies end of the string
    while(str[i] !='\0'){                                            // [1.5 mark]

        if(str[i]=='A' || str[i]=='E' || str[i]=='I' || str[i]=='O' || str[i]=='U'
        || str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u')// [5 mark]
            vCount++;                                                // [0.25 mark]
        else                                                         // [0.25 mark]
            cCount++;                                                // [0.25 mark]

        i++;                                                         // [0.25 mark]
    }

    printf("Number of Vowels in String: %d\n",vCount);               // [0.25 mark]
    printf("Number of Consonants in String: %d",cCount);             // [0.25 mark]
}

void removeNonAlphaCharacters(char line[100]){

    printf("Combined String: ");                                     // [0.25 mark]
    puts(line);                                                      // [0.25 mark]

    for (int i = 0, j; line[i] != '\0'; ++i) {                       // [2 mark]

        // enter the loop if the character is not an alphabet and not the null character
        while (!(line[i] >= 'a' && line[i] <= 'z') && !(line[i] >= 'A' && line[i] <= 'Z') &&
        !(line[i] == '\0')) {                                        // [6 mark]
            for (j = i; line[j] != '\0'; ++j) {                      // [3 mark]

            // if jth element of line is not an alphabet, then (j+1)th element = jth element
                line[j] = line[j + 1];                               // [2 mark]
            }
            line[j] = '\0';                                          // [1 mark]
        }
    }

    countVowelConsotant(line);                                       // [2 mark]
}
```

```c
void concatenate(char str1[50], char str2[50]){
    int i, j, len;

    // store the length of str1 in i
    for(i=0; str1[i]!='\0'; ++i);                                    // [2 mark]

    // concatenate the string str2 at the end of str1
    for(j=0; str2[j]!='\0'; ++j, ++i)                                // [2 mark]
        str1[i]=str2[j];                                             // [1 mark]

    // \0 represents end of string
    str1[i]='\0';                                                    // [1 mark]

    removeNonAlphaCharacters(str1);                                  // [1 mark]
}


int main(){
    char str1[50], str2[50];                                         // [1 mark]

    printf("Enter the first string: ");                              // [0.25 mark]
    fgets(str1, 50, stdin);                                          // [0.25 mark]
    printf("Enter the second string: ");                             // [0.25 mark]
    fgets(str2, 50, stdin);                                          // [0.25 mark]

    concatenate(str1, str2);                                         // [2 mark]

    return 0;                                                        // [0.25 mark]
}
```

Instructions:

- Correct passing of values to the *countVowelConsonant*, *removeNonAlphabetCharacters* and *concatenate* function. [6 mark]
- Use of *jal* and *jr* function to call functions and return from the functions respectively. [5 mark]

Following problem is focused on recursion in MIPS.

4. Write a complete running equivalent MIPS programs for the following C program: **[12 Marks]**

```
/*
Program description: Program to reverse a sentence using recursion
*/

#include <stdio.h>

void reverseSentence();

int main() {
    printf("Enter a sentence: ");        // [ 0.25 mark]
    reverseSentence();                   // [ 1 mark]
    return 0;                            // [ 0.25 mark]
}

void reverseSentence() {
    char c;                              // [ 1.5 mark]
    scanf("%c", &c);                     // [ 0.25 mark]

    if (c != '\n') {                     // [ 0.5 mark]
        reverseSentence();              // [ 5 mark]
        printf("%c", c);                // [ 0.25 mark]
    }
}
```

Instruction:

- Correct use of stack in calling the recursion. [3 mark]

Submission Guidelines:

1. All home assignments will be submitted **ONLY** through Canvas.

2. The program should be submitted in .asm file.
    a. Each of your .asm file need to have header information, i.e. it should include: full name, official email address, date and brief description of the program
    b. Make sure you use comments and appropriate label/variables name. Codes without comments will be given partial credit. Up to 50% of the marks would be deducted for not writing sufficient comments.

3. When naming your home assignments, please use the naming convention below:
    Assignment04_firstName_lastName_Question01.asm

4. All the codes should also be copied in textual form and save into pdf format. Please note that the code should not be copied in image form or saved as an image file.
    Assignment04_firstName_lastName_Question01.pdf

Failure to submit in this order will automatically results in 10 points deduction.