CS 1713
Introduction to Computer Programming II
Midterm 2

<table>
<thead>
<tr>
<th>Question</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1</td>
<td></td>
</tr>
<tr>
<td>Question 2</td>
<td></td>
</tr>
<tr>
<td>Question 3</td>
<td></td>
</tr>
<tr>
<td>Question 4</td>
<td></td>
</tr>
<tr>
<td>Question 5</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
</tr>
</tbody>
</table>

NAME:________________________

Instructions
1. Do all of the 5 problems
3. You have 70 minutes for the exam
4. Show all your work
5. Do not separate midterm papers

<table>
<thead>
<tr>
<th>Easy</th>
<th>Difficulty Level</th>
<th>Difficult</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐  ☐  ☐</td>
<td>☐  ☐  ☐  ☐  ☐  ☐  ☐  ☐  ☐  ☐</td>
<td>☐  ☐  ☐  ☐</td>
</tr>
</tbody>
</table>
1. (20 pts) What is the output of the following program? Show all your work.

```c
#include <stdio.h>

int function1(int data[10], int a, int b)
{
    return(data[b]-data[a-1]);
}

int main()
{
    int a[10];
    int i=1,x;
    a[0]=0;
    while (i<10)
    {
        a[i] = i+a[i-1];
        i++;
    }
    for (i=3; i<7; i++)
    {
        x = function1(a,i,i+2);
        printf("%d %d\n",i,x);
    }
}
```

```
3 12
4 15
5 18
6 21
```

```
0 1 3 6 10 15 21 28 36 45
0 1 2 3 4 5 6 7 8 9
```

```
3 2 1 0
5 4 3 2
6 5 4 3
```

```
3 5
12 15 18 21
```

```
3 5
4 6
5 7
6 8
```

```
\alpha \beta \gamma \delta \\
\alpha \beta \gamma \delta \\
\alpha \beta \gamma \delta \\
\alpha \beta \gamma \delta
```

```
return (data[5]-data[2])
```

```
4 6
```

```
return (data[6]-data[3])
```

```
15 17
```

```
return (data[7]-data[4])
```

```
6 8
```

```
return (data[8]-data[5])
```

```
21
```
2. (20 pts) What is the output of the following program? Show all your work.

```c
#include <stdio.h>

int function1(int *m)
{
    *m = *m + 1;
    return(*m);
}

int function2(int n)
{
    n = n + 2;
    return(n);
}

int main()
{
    int x = 2, y = 4;
    int *ptr = &x;

    printf("Output 1: X = %d, Y = %d\n", x, y);
    x = function2(y);
    printf("Output 2: X = %d, Y = %d\n", x, y);

    x = 2; y = 4;
    y = function1(ptr);
    printf("Output 3: X = %d, Y = %d\n", x, y);

    x = 2; y = 4;
    x = function2(function1(ptr));
    printf("Output 4: X = %d, Y = %d\n", x, y);

    x = 2; y = 4;
    x = function2(function2(y));
    printf("Output 5: X = %d, Y = %d\n", x, y);

    return 0;
}
```

Output 1: X = 2, Y = 4
Output 2: X = 6, Y = 4
Output 3: X = 3, Y = 3
Output 4: X = 5, Y = 4
Output 5: X = 8, Y = 4
3. (20 pts) C language does not provide a standard function that removing trailing spaces at
the end of a string. Write a function to do this. Sample executions of the function and the
function prototype are given below. You can use strlen() function in your implementation.
given below.
"Apple " becomes "Apple"
"An apple" becomes "An apple"
"Apple 2" becomes "Apple 2"

```c
char *strtrim(char *str)
{
    char *end;
    end = str + strlen(str) - 1;
    while (end >= str && isspace(*end))
        end--;

    *end+1 = '\0';
    return (str);
}
```

```c
In main
strtrim(str1)
```
4. (20 pts) Consider a coin system where there are coins for 1 cent, 4 cents and 16 cents instead of the 1 cent, 5 cents, 10 cents and 25 cents that we have now. Find out whether optimal change (minimum number of coins) for every amount from 1 cent to 60 cents can be given using at most 3 coins of each type in this system. Verify this using your program or print counterexample you find if there are any. Write a Complete program to do this. You can use functions in your program if you want.

```c
#include <stdio.h>

int findcoins(int n)
{
    int cl = 0, c4 = 0, c16 = 0, max = 0;
    cl = n / 16;
    c4 = (n - cl * 16) / 4;
    c1 = n - cl * 16 - c4 * 4;
    max = cl;
    if (c4 > max)
        max = c4;
    if (c1 > max)
        max = c1;
    return (max);
}

int main()
{
    int i, done = 0;
    for (i = 1; i <= 60 && done == 0; i++)
    {
        if (findcoins(i) > 3)
        {
            printf("You require more than 3 coins of each type w\n", i);
            done = 1;
        }
        if (done == 0)
            printf("Every case can be done with 3 coins of each type w\n");
    }
    return 0;
}
```
5. (20 pts) **Complete the following program** to find the first non-repeating character in a string. Read a string from the user and print the first non-repeating character. **Hint:** first non-repeating character is one of the characters in the string and appears in the string only once.

"appleforapple" has first non-repeating character 'f', since 'a', 'p', 'l', 'e' repeat
"an apple" has first non-repeating character 'n' since 'a' repeats
"abcdcdcb" has first non-repeating character 'a' since 'a' is first character and does not repeat

```c
#include <stdio.h>
#include <string.h>
// returns how many times ch appears in str
int countchar(char ch, char *str)
{
    int count = 0;
    while (*str != '\0')
    {
        if (*str == ch)
            count++;
        str++;
    }
    return(count);
}

int main()
{
    char str[100];
    char *str1;
    char norepeat;
    int done = 0;

    fgets(str,100,stdin);
    str1=str;

    while ((*str1 != '\0') && (done == 0))
    {
        if (countchar(*str1, str) == 1)
        {
            norepeat = *str1;
            done = 1;
        }
        str1++;
    }

    printf("First nonrepeating character is \%c\n", norepeat);
    return 0;
}
```