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

Easy

Difficulty Level

Difficult

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
1 2 3 4 5 6 7 8 9 10
1. (20 pts) What is the output of the following program? Show all your work.

```c
#include <stdio.h>

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

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

int main()
{
    int x = 1, y = 3;
    int *xptr = &x;
    int *yptr = &y;

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

    x = 1; y = 3;
    x = function1(xptr, x);
    printf("Output 3: X = %d, Y = %d\n", x, y);

    x = 1; y = 3;
    x = function2(function1(xptr, 1), yptr);
    printf("Output 4: X = %d, Y = %d\n", x, y);

    x = 1; y = 3;
    x = function1(yptr, function2(2, yptr));
    printf("Output 5: X = %d, Y = %d\n", x, y);

    return 0;
}
```
2. (20 pts) Write a function copynewstring to make a copy of a provided string and return a pointer to it. Note that although the contents of the returned string will be same as original string, they will be allocated in different parts of memory. Sample executions of the function and the function prototype are given below. Y

```
    copynewstring("An apple") returns the new string "An apple"
    copynewstring("Apple  2") returns the new string "Apple  2"
```

Complete the function below

```
char *copynewstring(char *str)
{
    int cnt = 0;
    char *str1 = str;
    while (*str1 != '\0')
    {
        cnt++;
        str1++;
    }
    str1 = (char *)malloc((cnt+1) * sizeof(char));
    cnt = 0;
    while (*str != '\0')
    {
        *(str1 + cnt) = *str;
        cnt++;
        str1++;
        *(str1 + cnt) = '\0';
    }
    return str1;
}
```

```
main
{
    str1 = copynewstring("Hello");
    str1 = str
```
3. (20 pts) What is the output of the following program? Show all your work. Draw the contents of the array and pay attention to the formatting of the output.

```c
#include <stdio.h>
#include <stdlib.h>

void mystery(int *dat, int n) {
    int i;
    for (i=0; i<n; i++)
        printf("%d ", dat[i]);
    printf("\n");
    return;
}

int main()
{
    int i,*data,*dataptr;
    data = (int *)malloc(5*sizeof(int));
    for (i=0; i<5; i++)
        data[i]=2*i;
    mystery(data,5);
    *data = 3;
    dataptr = data;
    dataptr++;
    *dataptr = 5;
    mystery(data,5);
    *(data+2) = 4;
    *(dataptr+2)=2;
    mystery(data,5);
    free(data);
    return 0;
}
```
4. (20 pts) Consider the following structure declaration for time. Write a function Add1Second that takes a structure as a parameter and adds 1 second to time represented by the structure.

```c
struct time
{
    int hour;
    int minute;
    int second;
};

typedef struct time time;
```

Sample execution of the function on times is given below. Make sure your function works for all of them.

- 8:52:53 becomes 8:52:54
- 2:34:59 becomes 2:35:00
- 10:59:59 becomes 11:00:00

Prototype of the function is given below. It has a single parameter.

```c
void add1second(  )
{
}
```
5. (20 pts) Write a function to find the middle node in a singly linked list and return a pointer to the middle node. If the list has $2k + 1$ elements where $k$ is an integer $\geq 0$, return the $(k + 1)^{th}$ node from the start. If the list has $2k$ elements where $k$ is an integer $\geq 0$, return the $(k + 1)^{th}$ node from the start. For the following linked list the function returns a pointer to node containing 12.

![Linked List Diagram]

List node definition is as follows

```c
struct node
{
    int info;
    struct node *next;
};
typedef struct node node;
```

Function prototype is given below.

```c
node *middleNode(node *list1)
{
    ...
}
```