EECE.2160 Spring 2018: Exam 3
Structure Definitions and Function Test Cases

Question 2a (`Res` and `ResPair` structure definitions; `setRes()` function definition)
```
typedef struct {
   double v;
   double i;
   double r;
} Res;

typedef struct {
   Res r1;
   Res r2;
} ResPair;

void setRes(Res *rptr, double rv, double ri, double rr) {
   rptr->v = rv;
   rptr->i = ri;
   rptr->r = rr;
}
```

Question 2b (`Box` structure definition)
```
typedef struct {
   double W;         // Width of box
   double L;         // Length of box
   double H;         // Height of box
} Box;
```

SEE OTHER SIDE FOR QUESTION 6 MATERIAL

EECE.2160 Spring 2018: Exam 3
Extra Credit Problem: Structure Definition and Additional Hints

Question 6 (`XCstruct` structure, additional hints)

```
typedef struct {
   unsigned nch;
   unsigned val;
   unsigned add[6];
} XCstruct;
```

Additional hints:

- Structure array initializations may look confusing because they're a combination of two things—structures and arrays—that use comma-separated lists enclosed in curly braces. Here's an example declaration to explain the basic idea:

  ```
  XCstruct arr[] = { {2, 3, {1} },   {5, 9, {-1, -10} };
  ```

  The array `arr[]` contains two structures, with the members of each as follows:

  - `arr[0].nch = 2`
  - `arr[0].val = 3`
  - `arr[0].add[]` is an array that holds the single value 1

  - `arr[1].nch = 5`
  - `arr[1].val = 9`
  - `arr[1].add[]` is an array that holds two values, -1 and -10

- You may find the following character subset from the ASCII value table useful. Note that, while not all letters are shown, the table is intended to show that capital letters cover the ASCII value range 65-90, while lowercase letters cover the ASCII value range 97-122. Consecutive letters have consecutive ASCII values—for example, since 'A' = 65, 'B' = 66, 'C' = 67, and so on.

Character	ASCII Value
Space	32
!	33
“	34
#	35

Character	ASCII Value
A	65
Z	90
a	97
‘z’	122