Homework #8 EEL 2880 – Structures

```
1.  /* Example: StructuresDirect in CodeBlocks
2.    structures as function arguments and return values
3.      Direct instances using names    */
4.
5.  #include <stdio.h>
6.  #include <time.h>
7.
8.  struct cplx {
9.                double real;  /* real part */
10.               double imag;  /* imaginary part */
11.             };
12.
13. struct cplx add(struct cplx a, struct cplx b);  /* function prototype */
14.
15.
16. int main(void)
17. {
18.   struct cplx x, y, z;
19.   time_t rawtime = time(NULL);
20.
21.
22.   x.real = 2.5;
23.   x.imag = 5.0;
24.   y.real = 3.2;
25.   y.imag = -1.7;
26.
27.
28.   z = add(x, y);
29.   printf("Direct Instance z = %4.2f + %4.2f j\n", z.real, z.imag);
30.   printf("Today is %s", ctime(&rawtime));
31.   return 0;
32. }
33.
34.
35. struct cplx add(struct cplx a, struct cplx b)
36. {
37.   struct cplx c = a;  /* can initialise an auto struct variable */
38.
39.   c.real += b.real;
40.   c.imag += b.imag;
41.   return c;           /* can return a struct value */
42. }
```

Illustration 1

Run the example program in Illustration #1.

1. Run the program from Illustration 1 and submit a copy of the console output.
2. Which lines form the prototype for a structure
3. What is the tag name of the structure
4. Which lines create instances of the structure, what are the structure names?
5. Which lines initialize the member elements of the structures? How?
6. Which line is the prototype for a function with structure parameters
7. Which lines are the function definition with structure parameters
8. How are the member elements of a named (direct instance) structure dereferenced (how are the values changed)?
9. How is the addition of two structures accomplished?

```
1.  /* Example: StructuresIndirect in CodeBlocks
2.    structures as function arguments and return values
3.      indirect instances using pointers    */
4.
5.  #include <stdio.h>
6.  #include <stdlib.h>
7.  #include <time.h>
8.
9.  struct cplx
10. {
11.     double real;  /* real part */
12.     double imag;  /* imaginary part */
13. };
14.
15. struct cplx add(struct cplx* pa, struct cplx* pb);  /* function prototype */
16.
17. int main(void)
18. {
19.     struct cplx z;
20.     time_t rawtime = time(NULL);
21.
22.
23.     struct cplx* px= malloc(sizeof(struct cplx));
24.     struct cplx* py= malloc(sizeof(struct cplx));
25.
26.     px->real = 2.5;
27.     px->imag = 5.0;
28.     py->real = 3.2;
29.     py->imag = -1.7;
30.
31.
32.     z = add(px, py);
33.     printf("Indirect Instance z = %4.2f + %4.2f j\n", z.real, z.imag);
34.     printf("Today is %s", ctime(&rawtime));
35.
36.     return 0;
37. }
38.
39.
40. struct cplx add(struct cplx* pa, struct cplx* pb)
41. {
42.     struct cplx c = *pa;  /* can initialise an auto struct variable */
43.
44.     c.real += pb->real;
45.     c.imag += pb->imag;
46.     return c;           /* can return a struct value */
47. }
```

Illustration 2

In C-language, a structure object can be created using the malloc () function.

See http://www.tutorialspoint.com/cprogramming/pdf/c_memory_management.pdf

So Illustration 2 creates instances of the structures while the program is running without a name but with a pointer to the memory space set aside.

10. Run the program from Illustration 2 and submit a copy of the console output.
11. Which lines create instances of the structure, what are the structure names?
12. What are the structure pointer names?
13. Which lines initialize the member elements of the structures? How?
14. Which line is the prototype for a function with structure pointer parameters
15. Which lines are the function definition with structure pointer parameters
16. How are the member elements of an unamed (indirect instance) structure dereferenced (how are the values changed)?
17. How is the addition of two structures accomplished?
18. Could a pointer be returned from the function?