Module 4
C, Objective C and SWIFT - MADP 401 (140 hours/7 weeks)

Assignment 7 (in-class activity)

Due: 5:30pm, Jan 31th, 2017

Requirements

- N/A

Problem1

- Create a structure called date:
  - Define a structure called date with three integer fields, day, month and year.
  - Declare a pointer to a structure of type *date* called *dates*.
  - Assign the value 10 to the field *day* using the *dates* pointer.
- Create a structure called machine
  - A structure of type *machine* contains two fields, an integer called *name*, and a char pointer called *memory*. Show what the definition of the structure looks like.
  - A pointer called *mpu641* of type machine is declared.
  - Assign the address of the character array *CPUtype* to the field *memory* using the pointer *mpu641*.
  - Assign the value 10 to the field *name* using the pointer *mpu641*.
- Create a structure called time:
  - A structure pointer *times* of type *time* (which has three fields, all pointers to integers, day, month and year respectively) is declared.
  - Using the pointer *times*, update the field *day* to 10.
  - An array of pointers (10 elements) of type *time* (as detailed above), called *sample* is declared.
  - Update the field *month* of the third array element to 12.
  - Declares a variable which is a pointer for each variable you declared above
  - Initializes the pointer to point at the variable
  - Use printf statement to print the values and address of each variable.
  - Add code that changes the value of the variable via the pointer (using the * symbol), then print out the pointers and values again to test your code.

Problem2

- Let's do some maths

- Define a structure called point
- Point has two integer values: x and y
- Define a function called distance, which takes two inputs of type point and calculates and returns the distance between the two points. The formula for calculating the distance between two points is: (x^2 + y^2)^1/2
- Define another function called printPoint, which takes a point and print its coordination in this format: the point is at coordinate [x,y] where x and y are the x and y coordination of the input point.
- Define another function called average which takes an array of points as well as a character. If the input charcter is 'x' then the function calculates the average of x coordination of all points in the array. Similarly if the input character is 'y' then it calculates the average of y coordination of the input points.
- Define another function called vectorAverage which takes an array of values of types point. The function then calculates the average x and average y using the average function you defined above and create another point whose x is equal to average x and y is equal to average y and returns the resulting point.
- Then use the printPoint function to print the resulting point from previous question.

Problem3

- A new data structure: Link List (Single linked list)
- Create a structure called item. The item has two variables: 1- one integer variable called age and one variable called next of type item.
- Create a function with no input but return a pointer to struct item.
- The function will prompt the user for up 10 times to enter an age.
- Every time the user enters an age, the function will create a value of type item and assign the entered age to the item's age variable.
- The item's nextItem variable is set to Null.
- In the item had been created before the current one, set the nextItem variable to point to the newly created item.
- After the user enters the 10 ages, use a for-loop to go through the array and print all ages stored in the link list.

Problem4

- Multiple Choice Questions: For each question, draw the debugging the table and memoty (if needed) on a piece of paper and find the answer. Then write the code and run it and verify your answer. The TA will check your debugging table and memory as well as the code.

- 4.1 What is the output of this C code?

```
#include <stdio.h>
struct student
{
    int no = 5;
    char name[20];
};
void main()
{
    struct student s;
    s.no = 8;
    printf("hello");
}
```

a) Nothing

b) Compile time error

c) hello

d) Varies

4.2- What is the output of this C code?

```
#include <stdio.h>
struct p
{
    int k;
    char c;
    float f;
};
int p = 10;
int main()
{
    struct p x = {1, 97};
    printf("%f %d\n", x.f, p);
}
```

a) Compile time error

b) 0.000000 10

c) Somegarbage value 10

d) 0 10

4.3- What is the output of this C code?

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

```
2.    struct student
3.    {
4.        char *name;
5.    };
6.    struct student s;
7.    struct student fun(void)
8.    {
9.        s.name = "newton";
10.       printf("%s\n", s.name);
11.       s.name = "alan";
12.       return s;
13.   }
14.   void main()
15.   {
16.       struct student m = fun();
17.       printf("%s\n", m.name);
18.       m.name = "turing";
19.       printf("%s\n", s.name);
20.   }
```

a) newton alan alan

b) alan newton alan

c) alan alan newton

d) Compile time error

4.4 What's the output of the following code?

```
1.    #include <stdio.h>
2.    struct temp
3.    {
4.        int a;
5.    } s;
6.    void func(struct temp)
7.    {
8.        s.a = 10;
9.        printf("%d\t", s.a); s
10.   }
11.   main()
12.   {
13.       func(s);
14.       printf("%d\t", s.a);
15.   }
```

a) 10 (Garbage Value)

b) 0 10

c) 10 0

d) (Garbage Value) 10

4.5 What is the output of this C code?

```
1.  #include <stdio.h>
2.  struct point
3.  {
4.      int x;
5.      int y;
6.  };
7.  void foo(struct point*);
8.  int main()
9.  {
10.     struct point p1 = {1, 2};
11.     foo(&p1);
12. }
13. void foo(struct point *p)
14. {
15.     printf("%d\n", *p->x++);
16. }
```

a) Compile time error

b) 1

c) Segmentation fault/code crash

d) 2

4.6- What is the output of this C code?

(Assuming size of int be 4)

```
1.  #include <stdio.h>
2.  struct temp
3.  {
4.      int a;
5.      int b;
6.      int c;
7.  } p[] = {0};
8.  main()
9.  {
10.     printf("%d", sizeof(p));
```

```
11.     }
```

a) 4

b) 12

c) 16

d) Can't be estimated due to ambigous initialization of array

4.7- What is the output of this C code?

```
1.      #include <stdio.h>
2.      struct student
3.      {
4.          char *name;
5.      };
6.      void main()
7.      {
8.          struct student s[2], r[2];
9.          s[1] = s[0] = "alan";
10.         printf("%s%s", s[0].name, s[1].name);
11.     }
```

a) alan alan

b) Nothing

c) Compile time error

d) Varies

4.8- What is the output of this C code?

```
1.      #include <stdio.h>
2.      struct point
3.      {
4.          int x;
5.          int y;
6.      };
7.      void foo(struct point*);
8.      int main()
9.      {
10.         struct point p1[] = {1, 2, 3, 4};
11.         foo(p1);
12.     }
13.     void foo(struct point p[])
14.     {
```

```
15.         printf("%d\n", p[1].x);
16.     }
```

a) Compile time error

b) 3

c) 2

d) 1

4-9 What is the output of this C code?

```
1.      #include <stdio.h>
2.      struct p
3.      {
4.          int x;
5.          char y;
6.      };
7.      void foo(struct p* );
8.      int main()
9.      {
10.         typedef struct p* q;
11.         struct p p1[] = {1, 92, 3, 94, 5, 96};
12.         foo(p1);
13.     }
14.     void foo(struct p* p1)
15.     {
16.         q ptr1 = p1;
17.         printf("%d\n", ptr1->x);
18.     }
```

a) Compile time error

b) 1

c) Segmentation fault

d) Undefined behaviour

4.10- What is the output of this C code?

```
1.      #include <stdio.h>
2.      struct student
3.      {
4.          char *c;
5.          struct student point;
```

```
6.      };
7.      void  main()
8.      {
9.            struct  student  s;
10.           s.c  =  "hello";
11.           printf("%s",  s.c);
12.     }
```

a) hello

b) Nothing

c) Varies

d) Compile time error

Bonus Problem (optional)

Look at problem 3 again. Now let's work on another useful data structure called Double Linked List. In Double Linked list there is one pointer to the next item and one to the previous item. Write a program which creates a Double Linked List and then write a function which takes the double linked list and sort it based on the variable age.