Discussion Lecture concepts review:

Please answer the following questions:

1. Structure
    a. Write the definition of a structure, whose name is S. The structure has two basic variables: an integer number named i, and a character named ch.
    b. Write the definition of two variables of structure type S whose names are s1 and s2.
    c. Write the C program statements to assign the i of s1 to be 0, and the ch of s2 to be 'a'.

```
struct S
{
        int i;
        char ch;
};

struct S s1, s2;

s1.i = 0;
s2.ch = 'a';
```

2. Union
    a. Write the definition of a union, whose name is U. The union has two mutually exclusive basic variables: an integer number named i, and a character named ch.
    b. Write the definition of two variables of union type U whose names are u1 and u2.
    c. Write the C program statements to assign the i of u1 to be 0, and the ch of u2 to be 'a', and the ch of u1 to be 'b'.

```
Union U
{
    int i;
    char ch;
};

union U u1, u2;
u1.i = 0;           /*the value of u1.ch is changed at the same time*/
u2.ch = 'a';        /*the value of u2.i is changed at the same time*/
```

3. Enumerator
    a. Write the definition of an enumerator whose name is Color. The enumerator has three members: green, red and blue. The value of the first member (green) is defined to be 2.
    b. Write the definition of two variables of enumerator type Color whose names are color1 and color2
    c. Write the C program statements to assign value of color1 to be red, and color2 to be green

```
enum Color
{
    green = 2,
    red,
    blue
};

enum Color color1, color2;
color1 = red;
color2 = green;
```

Discussion Think-Pair-Share Activity:

Extend the menu-driven floating-point calculator which three advanced functions.
Part2: Pi approximation. We will use monte-carlo method with “hit and miss” integration.

To design this program, let us take a look at the following questions before we start programing.

1. What is the input and output of this program for this assignment?
   Input: number of points, radius of the circle
   Output: the approximated Pi, the coordinates of all the numbers, messages on the screen.

2. To have one random point (x,y), how many random numbers shall we generate? How to generate random numbers whose range is [0, 2*r]?

   ```
   #include “time.h”
   …
   srand(0);
   …
   x = ((double)rand())/((double)RAND MAX)*s; /* s is the side of the square, i.e. 2*r */
   y = ((double)rand())/((double)RAND MAX)*s; /* s is the side of the square, i.e. 2*r */
   …
   ```

3. How can we test weather a point is in the circle or not?
   (x,y) is the coordinates of point P.
   if (x-r)*(x-r) + (y-r)*(y-r) <= r*r, P is in; otherwise, P is out.

4. How many variables will you need? What are they? What are the types? What are their scopes?

   A double variable for the radius, an unsigned int variable for the total number of points, an unsigned int variable for point counts, two double variables for the point coordinates, an unsigned int variable for the number of the inside points, …

   You can have as many variables as you need for this program.

5. The equation for Pi approximation is $4 * \frac{number\ o\ f\ points\ hitting\ circle\ area}{number\ of\ points\ hitting\ square\ area}$. What shall we be careful of for using this equation to get the approximate value of Pi? (HINT: *number o f points hitting circle area* and *number of points hitting square area* are integer numbers, Pi is supposed to be floating point)

   Integer division vs. floating point division

6. Which control flow will you use to implement this function?
   Loops with if-else inside …