COMP500 / ENSE501: Week 11 – Exercise:

EXERCISE NAME: *QSort Spheres*

Given the following source code:

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

// TODO: Declare a Sphere structure here:

float get_random_percent(void);
struct Sphere* generate_spheres(int how_many);
void print_spheres(struct Sphere* all_spheres, int size);

int main(void)
{
    srand((int)time(0));

    struct Sphere* the_spheres = 0;

    int running = 1;
    int number = 0;

    do
    {
        printf("Generate how many spheres? ");
        scanf("%d", &number);

        the_spheres = generate_spheres(number);

        printf("Before sorting:\n");
        print_spheres(the_spheres);

        // TODO: Call qsort to sort the spheres...

        printf("After sorting:\n");
        print_spheres(the_spheres);

        free(the_spheres);
        the_spheres = 0;
    }
    while (running);

    return 0;
}

// TODO: Define the get_random_percent function here:
// TODO: Define the generate_spheres function here:
// TODO: Define the print_spheres function here:
```

Implement the required source code for each **TODO** comment.

Start by declaring the **`Sphere`** structure. This structure must have three **`float`** members, a **`radius`**, a **`volume`**, and a **`surface_area`**.

Next, define the **`get_random_percent`** function such that it returns a random floating-point value between **`0.0f`** and **`1.0f`**, inclusive.

Next, define the **`generate_spheres`** function such that it creates values for **`how_many`** different spheres, saving each sphere's details into an element in an array of spheres stored on the heap. The function must then return the address of this heap allocation.

Each sphere must have a radius between **`0.0f`** and **`10.0f`** units. Use the **`get_random_percent`** function to generate a random radius for each sphere. The value returned from **`get_random_percent`** can be scaled by the **`generate_spheres`** function (times 10) to generate the random radius.

Also, based upon each sphere's radius, compute the sphere's volume and surface area.

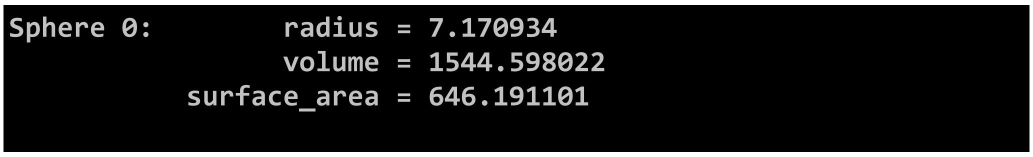
Remember, to calculate the volume of a sphere, use the following formula:

$$\textit{volume} = 4/3 \times \pi \times \textit{radius}^3$$

Remember, to calculate the surface are of a sphere, use the following formula:

$$\textit{surface\_area} = 4 \times \pi \times \textit{radius}^2$$

Next, define the **`print_spheres`** function such that it will print out the member data for each sphere in the **`all_spheres`** array, in the following style:



```
Sphere 0:          radius = 7.170934
                   volume = 1544.598022
             surface_area = 646.191101
```

Finally, define a comparison function to use with **`stdlib.h`**'s **`qsort`** function. The comparison function must be able to sort spheres by their **`radius`**. Next in **`main`**, call **`qsort`**.

Ensure the program output is exactly as described, and that the whitespace of your source code is well formatted. Utilise good naming practices when declaring variables.