Plagiarism in Computer Science

Plagiarism in Computer Science is typically defined as “either identical code, or code which only involves purely mechanical changes to a program, such as could be done by an automated system.” This includes renaming variables, reordering functions or procedures, or any other purely mechanical manipulation.

White space, tabbing, comments, and line spacing are not considered to be part of the “active” code, and will be ignored when examining programs for evidence of plagiarism.

As an example, consider the original code below. The first is the “original” code, the second is an example of the types of manipulations that could constitute plagiarism. This code is not intended to be useful code, it simply serves as an example.

Original Code:

```
public vacuumworld (int rooms, int percentdirty, int startroom)
{
	int numdirty=0;
	dirty = new boolean[rooms];
	this.room = 0;
	this.numrooms = rooms;
	for (int i=0; i<numrooms; i++)
	{
		if ((generator.nextInt(100)+1) > percentdirty)
		{
			dirty[i] = false;
		}
		else
		{
			dirty[i]=true;
			numdirty++;
		}
	}
	if ((startroom <= -1) || (startroom > numrooms))
		currentroom = generator.nextInt(numrooms);
	else
		currentroom = startroom;

	System.out.printf ("    **WORLD CREATED. %d ROOMS\n", numrooms);
	for (int i=0; i<rooms; i++)
	{
		System.out.printf ("    ***ROOM %d: ", i);
		if (dirty[i])
			System.out.printf ("DIRTY\n");
		else
			System.out.printf ("CLEAN\n");
	}
	PerfectScore = numdirty * 5;
}
```

```
public vacuumworld (int rooms, int percentdirty, int startroom)
{
        this.numrooms = rooms;
        dirty = new boolean[rooms];

        this.myroom = 0;
        int mynumdirty=0;

        for (int i=0; i<numrooms; i++)
        {
                if ((percentdirty <= generator.nextInt(100)+1))
                {
                        dirty[i] = false;
                }
                else
                {
                        dirty[i]=true;
                        numdirty++;
                }
        }
        if ((startroom > numrooms) || (startroom <= -1))
                currentroom = generator.nextInt(numrooms);
        else
                currentroom = startroom;

        System.out.printf ("    My world has started, with %d rooms\n", numrooms);
        for (int i=0; i<rooms; i++)
        {
                System.out.printf ("    ***ROOM %d: ", i);
                if (dirty[i])
                        System.out.printf ("DIRTY\n");
                else
                        System.out.printf ("CLEAN\n");
        }
        PerfectScore = (numdirty * 4) + (numdirty * 1);
}
```

This is a re-ordering of the variables.

This is a renaming of the variables.

Here, we have substituted an equivalent equation

This is a reordering of instructions. Reordering of functions falls into the same category.

Here, the order of the comparison has changed (which has no effect on the program)

The print statement has been changed

Another substitution of an equivalent equation.