Lab 11

Due: Thursday, April 22, 2021, by the end of your lab section.

You must submit lab through Canvas.

Instructions:

Listed below is a series of short videos posted by the School of Computing’s very own Dr. Jacob Sorber. Dr. Sorber is a great instructor and very knowledgeable in the C programming language. If you have the opportunity to take him for Operating Systems you should do so.

You are required to watch Dr. Sorber’s “Code Smells” videos. There are 5 videos in this series. A link to each video is listed below along with a set of questions that you must answer for each video.

Your answers MUST be in RED. If your answers are not in red points will be deducted.

These videos are short but very informative.

Code Smells

**Segment 1 – Smelly code and Magic Numbers.**

<https://www.youtube.com/watch?v=p8RC_i9t0MU>

According to Dr. Sorber, what is a magic number?

“magic numbers are just numbers” – Dr. Sorber. Numerical constants that should probably be replaced by a symbolic representation.

Can you think of an example of a magic number.

A great example is from a program that I wrote not half an hour ago. I had the number 5 in a place representing the length of a string, since my instructor told us that our program was only going to deal with 5-character strings. After realizing that this may be vague to a reader, I changed it to “<string\_name>.size()” instead (we are using C++) so that it would be more versatile and easier to understand.

What is the problem with Magic Numbers. He gives 2 problems what are they.

1. They are difficult for the reader to figure out
2. They are difficult to change later down the road

Dr. Sorber gave two reasons when he believes it is acceptable to use a magic number. Name one.

Magic numbers are okay when using a counter from or to 0, since that is easy to understand.

Dr. Sorber indicates using magic numbers is not always bad and discussed things you should consider when making your decision, with respect to the use of magic numbers. What where they?

1. Will this code be easier to read without magic numbers?
2. Does removing magic numbers make this code easier to change and maintain?

Code Smells

**Segment 2 – Duplicate Code**

<https://www.youtube.com/watch?v=ck_RfVOYgjQ>

Why is duplicate code not your friend?

It makes your code longer, creates more opportunities for bugs, makes code harder to maintain, and makes code harder to change. Similarly to magic numbers, it’s easier to change one bit of code than a bunch.

In the videos, Dr. Sorber introduces the D.R.Y. principle. What does D.R.Y. stand for and what is his solution to this problem.?

Don’t Repeat Yourself. His solution is to create a function or macro that you could call instead of copy/pasting the section of code.

Dr. Sorber states, the D.R.Y. principle makes your code shorter, easier to read, maintain, and more likely to make your coworkers not hate you.

Code Smells

**Segment 3 Bad Names**

<https://www.youtube.com/watch?v=zx7euEEZ0H4>

Dr. Sorber mentioned several problems that can arise if you do not practice good naming conventions in your code. What where some of the issues that may arise for poor naming convention.

They can be hard to understand, inconsistent, and long.

He listed some naming conventions that beginner students use that may or may not always be acceptable.

Name at least 5 he listed and give an example of each. Do not just copy his example, think about these and give your own example.

1. Names that mean nothing: foo, bar, a, b, c, etc
2. Names that sound good but mean the wrong thing: vector<int> solutions could be a list of solutions to a problem but it could also be used as a stack or queue in a program. (I have made this mistake before)
3. Snake case: this\_name\_uses\_snake\_case
4. Camel case: thisNameUsesCamelCase
5. Screaming snake case: THIS\_NAME\_USES\_SCREAMING\_SNAKE\_CASE

When is it appropriate **not** be concerned with your naming convention and when **should** you be concerned with the naming convention?

It is appropriate to not be concerned when you are the only one who will read or use your code. Otherwise, you should follow proper naming convention so that others can understand your code.

When choosing names for your code Dr. Sorber listed 3 things you need to think about. List the three in their order of importance and give a short description of each.

1. Understandable names: Names should describe what the variable is used for in a concise and easy to understand way
2. Consistent names: Naming convention should be consistent so as not to confuse the reader
3. Short names: Easier to type, and quicker to read

What are some of the examples of multiple word naming convention he mentioned?

1. snake\_case
2. camelCase
3. SCREAMING\_SNAKE\_CASE

List the one that most reflects your style of naming convention.

I usually use camel case for local variables and screaming snake case for #define’d variables, global constants, etc.

Code Smells

**Segment 4 Comments**

<https://www.youtube.com/watch?v=LLqDNjr0kPo>

This video covers the problem with having two few comments and to many comments.

Dr. Sorber listed a really interesting Truth about bad code and comments. What was that Truth?

Comments are morally neutral. Too few comments is just as bad as too many comments.

Why does Dr. Sorber believe comments exist?

Comments exist to help a reader understand what the code does

Why do they not exist?

Comments do not exist to compensate for bad coding style.

What did Dr. Sorber state was your first priority when writing code and what makes good code.

Your first priority should be writing good code. This includes variable names, function names, duplicate code, and magic numbers. Comments don’t exist to fix these problems, they exist to enhance good code.

Dr. Sorber states having a lot of comments is not a bad thing but not always necessary. He discussed problems with requiring big comment blocks, what did he discuss?

Large comment blocks must stay up to date with the code. If code is changed without updating comments, then the comments become a hinderance instead of a helper.

Code Smell’s

**Segment 5 – Long functions**

<https://www.youtube.com/watch?v=ll4XT0MYKN0>

Discuss the problem with really long functions.

You can’t keep long functions on the screen at once, so therefore it is harder to keep the whole function in your head at once. Lots of scrolling is also mentally exhausting.

In Dr. Sorber’s opinion, what is the rule of thumb with respect to the size of a function?

A function should usually be able to fit entirely on the screen.

Dr. Sorber discussed another rule that could help keep your functions short and simple. What is that rule?

Your functions should be able to be summarized in one sentence.