UNIT 5 CODING PRACTICES

Change at RKU

AGENDA of CODING PRACTICES

- Programming principles and guidelines.
- Programming practices.
- Coding standards.
- Incremental development of code

Programming **Principals** and Guidelines

- The principles of good programming are closely related to principles of good design and engineering.
- **DRY - Don't repeat yourself** - This is probably the single most fundamental tenet (law) in programming is to avoid repetition. Many programming constructs exist solely for that purpose (e.g. loops, functions, classes, and more). As soon as you start repeating yourself (e.g. a long expression, a series of statements, same concept) create a new abstraction.

Continue…

- **Abstraction Principle** - Related to DRY is the abstraction principle “Each significant piece of functionality in a program should be implemented in just one place in the source code.”
- **KISS (Keep it simple, stupid!)** - Simplicity (and avoiding complexity) should always be a key goal. Simple code takes less time to write, has fewer bugs, and is easier to modify.

Continue…

- **Avoid Creating a YAGNI (You aren't going to need it)** - You should try not to add functionality until you need it.
- **Do the simplest thing that could possibly work** - A good question to ask one's self when programming is "What is the simplest thing that could possibly work?" This helps keep us on the path towards simplicity in the design.

Continue…

- **Don’t make me think** - The point is that code should be easily read and understood with a minimum of effort required. If code requires too much thinking from an observer to understand, then it can probably stand to be simplified
- **Single Responsibility Principle** - A component of code (e.g. class or function) should perform a single well defined task.

Continue…

- **Minimize Coupling** - Any section of code (code block, function, class, etc) should minimize the dependencies on other areas of code. This is achieved by using shared variables as little as possible. “Low coupling is often a sign of a well-structured computer system and a good design

Programming Principals and **Guidelines**

- Writing an efficient software code requires a through knowledge of programming. This knowledge can be implemented by following a coding style which comprises several guidelines that help in writing the software code efficiently and with minimum errors. These guidelines, known as **coding guidelines,** are used to implement individual programming language constructs, comments, formatting, and so on. These guidelines, if followed, help in preventing errors, controlling the complexity of the program, and increasing the readability and understandability of the program.

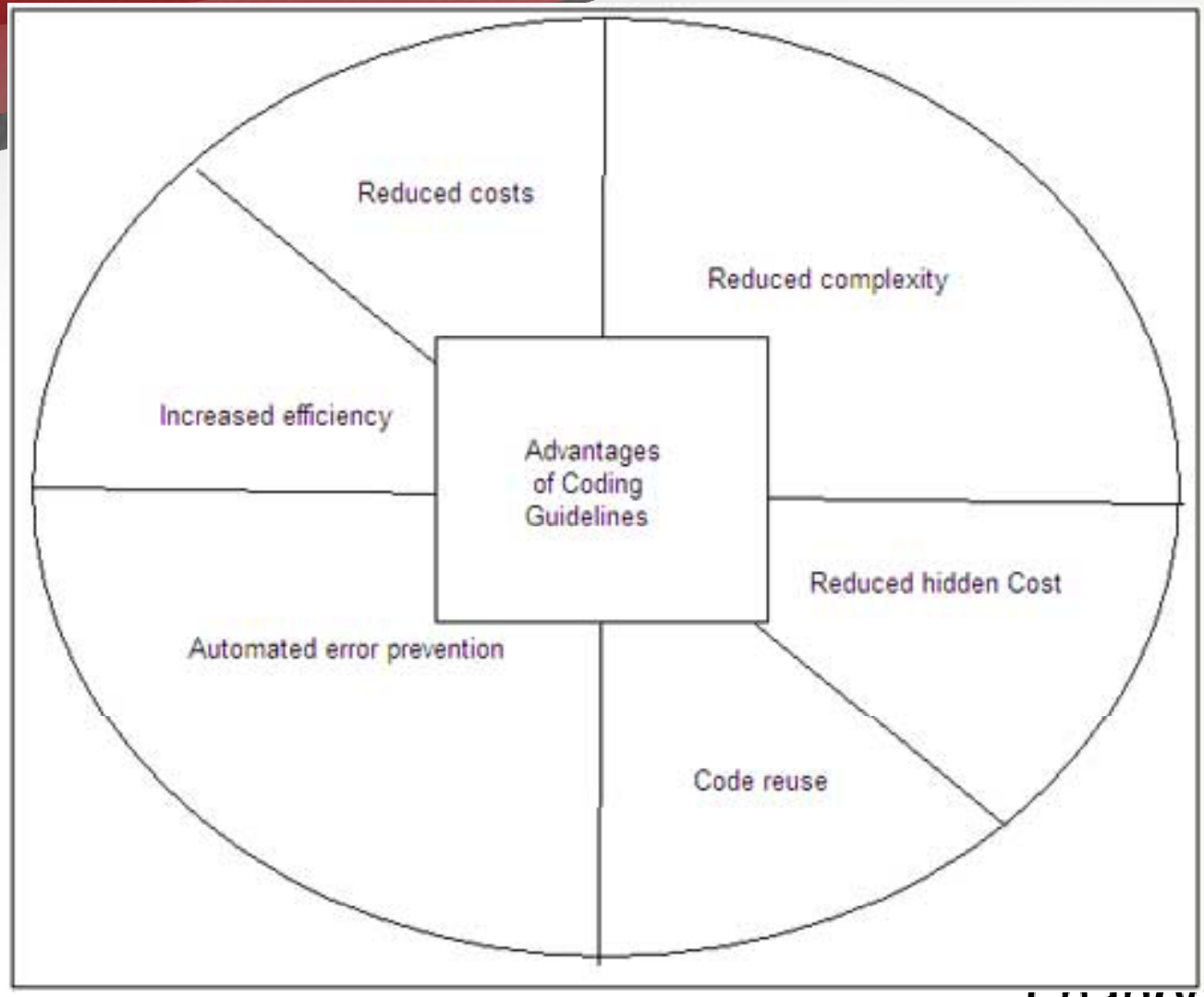
Continue…

- There are certain guidelines for naming variables, functions and methods in the software code. These naming conventions help software; developers in understanding the use of a particular variable or function. The guidelines used to assign a name to any variable, function, and method are given below.

Continue…

- All the variables, functions, and methods should be assigned names that make the code more understandable to the reader. By using meaningful names, the code can be self-explanatory, thus, minimizing the effort of writing comments for variables.

Advantage of Coding Guidelines



Coding Standards

- **Commenting**
- Name of the module
- Purpose of the Module
- Description of the Module
- Original Author
- Modifications
- Authors who modified code with a description on why it was modified.

- **Naming conventions**
- **Keep the code simple**
- **Portability**

- A general overview of all of the above:
- Know what the code block must perform
- Maintain naming conventions which are uniform throughout.
- Indicate a brief description of what a variable is for (reference to commenting)
- Correct errors as they occur.
- Keep your code simple