SHIKSHA BHARATI GLOBAL SCHOOL

Sector-8, Dwarka, New Delhi-110077

SKILL MODULE

Subject: Coding

Class- VI and VII

Coding is a creative activity that students from any discipline can engage in. It helps to build computational thinking, develop problem solving skills, improve critical thinking and exposure to real life situations to solve problems in various realms.

Therefore, CBSE is introducing 'Coding' as a skill module of 12 hours duration in classes VI-VIII from the Session 2021-2022 onwards. The idea is also to simplify the coding learning experience by nurturing design thinking, logical flow of ideas and apply this across the disciplines. The foundations laid in the early years will help the students to build the competencies in the area of AI, data sciences and other disciplines.

CBSE acknowledges the initiative by Microsoft India in developing this coding handbook for class VI students. This handbook introduces concepts of coding and computational thinking using real life examples and block coding with open source Make Code platform. It uses gamified learning approach to make learning experience more engaging. The book is intuitive with practical examples of theoretical concepts and applied exercises. There are mini projects that students can work on. Additionally, the handbook also focuses on creating exposure to ethics of coding and promotes empathy among students by activities curated to demonstrate empathy and sensitivity.

The purpose of the book is to enable the future workforce to acquire coding skills early in their educational phase and build a solid foundation to be industry ready.

Here's a more detailed look at what a Coding skill module might entail:

Key Components of Coding Skill Module:

1. Readability

Readability ensures that code is easy to understand and follow. This is crucial for collaboration among developers and long-term project sustainability. Good readability includes:

- Clear and meaningful variable, function, and class names.
- Proper indentation and formatting.
- Avoiding overly complex logic and deeply nested structures.
- Adding comments where necessary but avoiding redundant or obvious ones.

Readable code reduces the learning curve for new developers and makes debugging and maintenance more efficient.

2. Maintainability

Maintainability refers to how easily code can be updated, modified, and extended over time. To achieve high maintainability, developers should:

- Follow the Single Responsibility Principle (SRP) and modularize code.
- Reduce tight coupling between components.
- Write reusable and loosely coupled functions and classes.
- Adhere to coding standards and best practices.
- Apply the DRY (Don't Repeat Yourself) principle, which reduces redundancy and ensures that functionality is written once and reused where necessary.
- Follow the KISS (Keep It Simple, Stupid) principle, which encourages developers to write simple and straightforward code rather than over-engineering solutions.

Well-maintained code allows teams to adapt to changes, fix bugs, and implement new features with minimal effort.

3. Efficiency

Efficient code optimizes performance and resource usage, ensuring the software runs smoothly under various conditions. Best practices for efficiency include:

- Choosing appropriate data structures and algorithms.
- Minimizing unnecessary computations and memory consumption.
- Optimizing database queries and caching mechanisms.
- Using concurrency and parallelism when necessary.

Efficiency is critical for applications that require high-speed execution, such as real-time systems, web services, and mobile applications.

4. Reliability

Reliability ensures that code functions correctly under expected conditions and gracefully handles unexpected situations. To improve reliability, developers should:

- Write comprehensive unit and integration tests.
- Handle errors and exceptions properly.
- Implement logging and monitoring for debugging and performance analysis.
- Test edge cases and stress conditions.

Reliable code reduces downtime, improves user experience, and builds trust in the software.

5. Security

Security protects code from vulnerabilities and threats, ensuring data integrity and privacy. Secure coding practices include:

- Validating and sanitizing user inputs to prevent injection attacks.
- Implementing authentication and authorization mechanisms.
- Encrypting sensitive data in transit and at rest.
- Regularly updating dependencies and fixing security vulnerabilities.

Security is an ongoing concern, and developers must stay vigilant to evolving threats and best practices.

Module Link:

For Class-VI
https://cbseacademic.nic.in/web_material/codeingDS/classVI_Coding_Student_Handbook.pdf

For Class-VII
https://cbseacademic.nic.in/web_material/codeingDS/classVII_Coding_Student_Handbook.pdf