Discussion-1
Module-1: Introduction to Programming

Discussion Topic:

What are the principle components involved in developing a program? Describe the benefits of breaking up a program into small modules and provide an example of an application that makes the most of one of these benefits.

My Post:

Hello class,
Historically, programs were available to individual and small businesses, on Compact Disks (CDs), and maintenance (upgrading/updating) was largely available only to larger organizations that could afford a programming team or for-hire software company. Today, in contrast, modern software development generally follows a cyclical programming model, and is accessible to everyone, often for a subscription fee. The cycle usually compromises planning and analysis, design, implementation, testing, and maintenance components. (Ricciardi, 2024)

The components can be listed as follows:

1. Analyze the problem or need: Determine the nature of the issue or requirement and devise a programmatic solution.
2. Design the program (logic): Employ tools such as flowcharts to map out the program structure.
3. Code the program: Develop the program using an appropriate programming language.
4. Debug the program (test): Locate and resolve any programming errors or bugs.
5. Formalize the solution: Finalize the program for release and complete the necessary documentation for its use and future upkeep.
6. Release and maintain the program.

(Nguyen, 2019)

The practice of breaking up a program into small modules (modular programming) can be referred to as Modular Programming. Modular programming is a technique that involves dividing a program into small-separate modules that can operate independently and can be interchangeable (Busbee & Braunschweig, 2018).

This approach has many benefits such as:

1. Readability: Modular code improves readability and comprehension by segmenting the code into clear, logical sections that each carry out a specific function.
2. Manageability: The use of smaller, self-contained modules simplifies management because alterations in one module generally do not affect others.
3. Easier Testing: Independent testing of modules facilitates the straightforward isolation and correction of errors.

4. Reusability: Components such as functions or classes from one module can be utilized in different sections of the program or in subsequent projects. This can reduce the time needed for development.
5. Maintenance and scalability: This is updating a module to meet new requirements, scale the program to add functionality, or fix bugs. This improves over time the reliability of the program and longevity of the program, as well as its functionality and scalability.

(Busbee, 2013, Preface)

Java programming implements modularity. Programs are built by combining classes, and every class can be treated as a small module; furthermore, each public class is stored in its own Java file. For instance, the Java Platform uses the Java Runtime Environment (JRE) and Java Development Kit (JDK), which are built to support these modular components. Starting with Java 9, the core runtime has been modularized, allowing developers to pick and choose which components to package with their applications, significantly simplifying the implementation and testing phases (Gassner, 2018).

However, breaking programs into smaller modules has some disadvantages, such as:

1. Increased complexity: Breaking a program into smaller modules can make it more complex and difficult to understand for new developers.
2. Higher cost: Modularization can be expensive, requiring more time and resources to design and implement.
3. Integration issues: Integrating small modules into larger modules, and large modules into even larger modules can be difficult, and time-consuming, it requires as well significant troubleshooting effort to ensure that all modules are compatible.
4. Performance overhead: Modules can influence performance, adding overhead, as they require additional computation and memory usage to communicate between modules.

(Ramadan, 2023)

In conclusion, breaking programs into smaller modules can bring many benefits to software development. Benefits, such as improved maintainability, reusability, and scalability. However, it also has its disadvantages, such as increased complexity, higher cost, and performance overhead. Thus, when developing an application, it is important to carefully consider the disadvantages and advantages of implementing modularizing while considering the specific needs of your project.

An application that benefits from modular programming is a large-scale web application like an online retail platform (e.g. Amazon). Such platforms require handling vast amounts of user and product data, secure transaction processing, and the ability to scale resources dynamically based on demand. Using a modular programming approach, coupled with a microservices architecture (a way to develop applications by breaking them into smaller parts, each with its own responsibilities (Google Cloud, n.d.), allows such complex systems to remain manageable and scalable. Each module, or service, can be developed, deployed, and scaled independently, facilitating more efficient updates and maintenance (Struk, 2023).

Side note: That is a long post... Not sure if it is a good or a bad thing?

-Alex

References:

Busbee, K. L. (2013, January 10). *Programming fundamentals - A modular structured approach using C++.* Internet Archive. https://archive.org/details/cnx-org-col10621/page/n5/mode/2up

Busbee, K., & Braunschweig, D. (2018, December 15). *Modular programming*. Programming fundamentals. https://press.rebus.community/programmingfundamentals/chapter/modular-programming/

Gassner, D. (2018, January 19). *Java 8+ Essential training: Syntax and structure.* Principles and components of Java [Video]. LinkedIn Learning. https://www.linkedin.com/learning/java-8-plus-essential-training-syntax-and-structure/principles-and-components-of-java?resume=false&u=2245842Links to an external site.

Google Cloud, (n.d.). *What is Microservices Architecture?* Google. https://cloud.google.com/learn/what-is-microservices-architecture#:~:text=Microservices%20architecture%20(often%20shortened%20to,its%20own%20realm%20of%20responsibilityLinks to an external site..

Nguyen, P. (2018). *Programming development cycles*. California State University, Long Beach. https://home.csulb.edu/~pnguyen/cecs100/lecturenotes/Programming%20Development%20Cycles.docxLinks to an external site.

Ramadan, M. (2023, February 3). Unlocking the advantages and disadvantages of modular architecture in software development. *Medium*. https://medium.com/@mohamedramadan_18266/unlocking-the-advantages-and-disadvantages-of-modular-architecture-in-software-development-38b6b3cf7ace#:~:text=Increased%20complexity%3A%20Breaking%20a%20system,resources%20to%20design%20and%20implement.

Ricciardi, A. (2024, February 11). *Module 1: Discussion Forum* [Post]. Information Technology: ITS320 – Basic Programming, Colorado State University Global.

Struk, S. (2023, October 17). *Decoding the complexities of large-scale web application development*. Relevant Software. https://relevant.software/blog/guide-large-scale-web-application-development/