CS 4402-01 Comparative Programming Languages

Instructor: Professor Rajesh Soni

Name: Ryohei Hayashi

Learning Journal 3

**Overview of the Week**

This week, we focused on learning about elementary data types, composite data types, and the concepts of variables, binding, and scope in programming languages. Regarding elementary data types, we covered topics such as encoding ASCII and UNICODE, integer operations (including addition, subtraction, multiplication, division, and overflow), enumeration types, character types, and boolean types. A particular emphasis was placed on understanding the evaluation and order of operations in expressions and the mechanics of assignments.

In composite data types, we explored records, arrays, references, string types, multidimensional arrays, bitwise computation, and the concepts of big-endian and little-endian memory representation. We also delved into their implementation methods and the significance of each concept.

The learning objectives included defining and understanding the implementation of these data types, expressions, and assignments within programming languages, as well as grasping the importance of bitwise operations and endian concepts. Additionally, the discussion assignments provided opportunities to apply these concepts, particularly in the contexts of overflow and endian.

**Personal Reflections**

This week’s topics emphasized the foundational understanding of data types and their role in programming, which I found crucial for building more complex programming skills. Learning about overflow and its potential security risks was particularly eye-opening, as it highlighted the importance of careful data management in program design. Similarly, understanding how endian differences affect data processing and file formats gave me a clearer perspective on their real-world implications.

However, I found it challenging to fully comprehend the practical applications of bitwise computation and multidimensional arrays. The endian concepts, while theoretically clear, were difficult to visualize in practical scenarios. I plan to deepen my understanding by working on hands-on coding examples in the future.

**Topics Studied in Depth**

This week, I focused in depth on the concepts of endianness and overflow. Regarding endianness, I learned how big-endian and little-endian formats impact memory representation of data. Big-endian is more intuitive and human-readable, while little-endian is efficient for specific architectures. Understanding the advantages and disadvantages of each approach helped me grasp how they are applied in different contexts.

For overflow, I explored its occurrence in integer operations and its implications for security. Through internet research, I learned about how overflow-related vulnerabilities, such as buffer overflow attacks, can compromise system security. This deepened my understanding of the importance of implementing safeguards in programming. Overall, these insights emphasized the connection between theoretical knowledge and its practical applications in programming.

**Future Challenges and Goals for Next Week**

Next week, we will learn about control structures, including sequences, selection, and repetition, as well as subprograms, recursion, and stack architecture. These concepts are essential for managing the flow of execution in programs, and I anticipate that recursion and stack architecture will be particularly challenging. Additionally, understanding the differences between call-by-value and call-by-reference mechanisms will be a key focus.

As a future challenge, I aim to solidify my practical skills in applying this week’s concepts, particularly bitwise operations and endianness. To achieve this, I plan to write sample programs to experience these concepts in action. Additionally, I will review the basics of recursion and subprograms in preparation for next week’s learning.

Word Count: 517

References

1. Ben-Ari, M. (2006). *Understanding programming languages.* Weizman Institute of Science.