20CYS312 - Principles of Programming Languages
Exploring Programming Paradigms

Assignment-01

Presented by Vinayak R
CB.EN.U4CYS21084
TIFAC-CORE in Cyber Security
Amrita Vishwa Vidyapeetham, Coimbatore Campus

Feb 2024

Outline

1. Paradigm 1 - Procedural
2. Pascal
3. Paradigm 2 - Dataflow
4. Blender Game Engine
5. Comparison and Discussions
6. Bibliography

Paradigm 1 - Procedural

- Procedural programming is centered around a series of instructions provided to the computer. These sets of instructions are commonly referred to as **procedures**. Procedural programming is considered a subset of the **Imperative Programming Paradigm**.
- Procedural programming adopts a **top-down approach** due to the execution of instructions in well-defined sequences.
- In procedural programming, our instructions are organized into smaller blocks of clearly defined code. These blocks of code are referred to as **functions**, with each function carrying out a distinct task. Executing a procedural program involves calling the appropriate function at the right time, specifically using the suitable function at the correct position within the extensive code. These functions often receive data and manipulate it to generate an output.

Paradigm 1 - Procedural

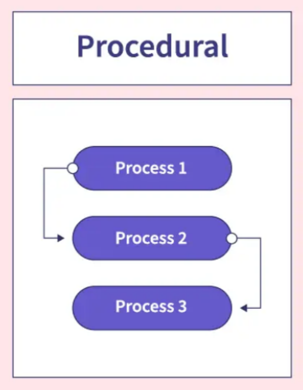


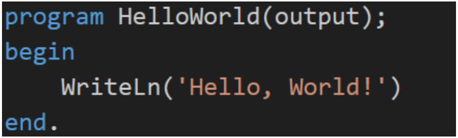
Figure: Procedural Programming

Pascal

- Pascal is an imperative and **procedural programming** language intended to encourage good programming practices using structured programming and data structuring.
- Pascal programs start with the *program* keyword with a list of external file descriptors as parameters; then follows the main block bracketed by the *begin* and *end* keywords. *Semicolons* separate statements, and the *full stop* ends the whole program. *Letter case* is ignored in Pascal source.

Pascal

- **Example Code:**



```
program HelloWorld(output);
begin
    WriteLn('Hello, World!')
end.
```

Figure: Pascal - Hello World Program

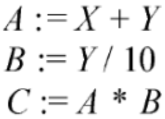
The text "Hello, World!" is printed as the output of the execution of above code.

Features of Pascal Language

- Pascal is a strongly typed language.
- It offers extensive error checking.
- It offers several data types like arrays, records, files and sets.
- It offers a variety of programming structures.
- It supports structured programming through functions and procedures.

Paradigm 2 - Dataflow

- Dataflow programming (DFP) is a programming paradigm where program execution is conceptualized as data flowing through a series of **operations** or transformations. Each operation may be represented as a node in a graph. **Nodes** are connected by directed arcs through which data flows. A node performs its operation when its input data are available. It sends out the result on the output arcs. Dataflow programming aids **parallelization** without the added complexity that comes with traditional programming.
- **Simple program and its dataflow equivalent:**



```
A := X + Y
B := Y / 10
C := A * B
```

Figure: A simple program

Paradigm 2 - Dataflow

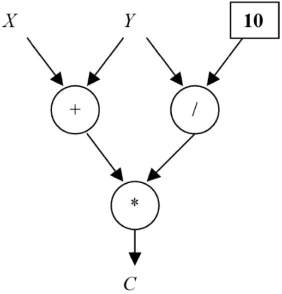


Figure: Dataflow equivalent

Blender Game Engine

- In the context of game engines, Dataflow programming can be used to design game logic in a visual and intuitive way. This allows developers to create complex behaviours without necessarily writing code.

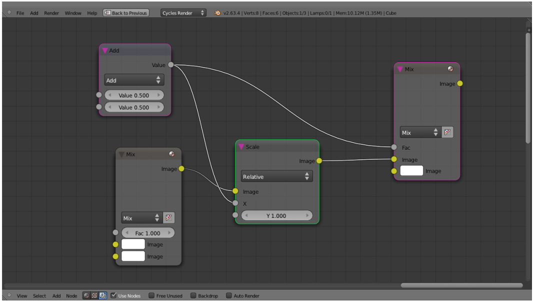


Figure: BGE

Blender Game Engine

- In Blender Game Engine, **logic bricks** were used as a visible programming interface. These logic bricks are visually connected to create a flow where the character moves only when a specific key is pressed.
    - **Sensor Brick (Input):** Detects a key press.
    - **Controller Brick (Processing):** Performs a logical operation (e.g., AND, OR).
    - **Actuator Brick (Output)**: Moves a character.
- Blender provides a number of different editors for displaying and modifying different aspects of data.
    - Text Editor
    - Node Editor
    - Logic Editor

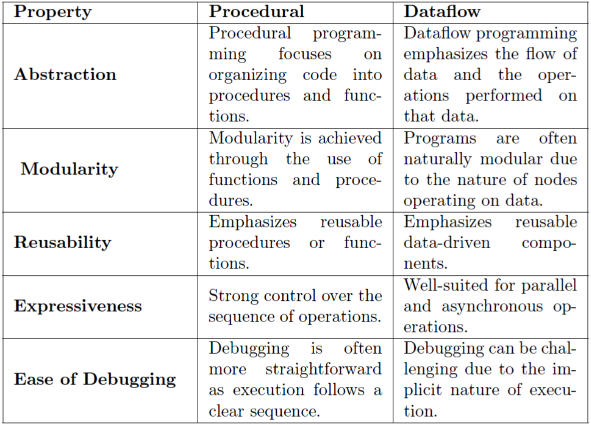
Comparison and Discussions

Similarities between Procedural and Dataflow

- Both dataflow programming and procedural programming are forms of imperative programming, where the focus is on defining a series of steps to accomplish a particular goal.
- Both paradigms involve the transformation of data.
- Both paradigms support the encapsulation of logic.
- Both paradigms encourage modularity.
- Both paradigms can be designed to be stateful.

Comparison and Discussions

Differences between Procedural and Dataflow



Property	Procedural	Dataflow
Abstraction	Procedural programming focuses on organizing code into procedures and functions.	Dataflow programming emphasizes the flow of data and the operations performed on that data.
Modularity	Modularity is achieved through the use of functions and procedures.	Programs are often naturally modular due to the nature of nodes operating on data.
Reusability	Emphasizes reusable procedures or functions.	Emphasizes reusable data-driven components.
Expressiveness	Strong control over the sequence of operations.	Well-suited for parallel and asynchronous operations.
Ease of Debugging	Debugging is often more straightforward as execution follows a clear sequence.	Debugging can be challenging due to the implicit nature of execution.

References

- https://devopedia.org/dataflow-programmingqst-ans-6
- https://www.youtube.com/watch?v=iFIT93wakVot=362s
- https://www.youtube.com/watch?v=yKKy0vJ0CrY
- https://docs.blender.org/manual/en/latest/
- https://blenderartists.org/t/design-for-a-new-graphical-game-engine/455219
- https://programiz.pro/resources/what-is-procedural-programming/
- https://www.geeksforgeeks.org/introduction-of-programming-paradigms/