Amrita Vishwa Vidyapeetham
TIFAC-CORE in Cyber Security

20CYS312 - Principles of Programming Languages Assignment-01: Exploring Programming Paradigms

G Manomithran

21st January, 2024

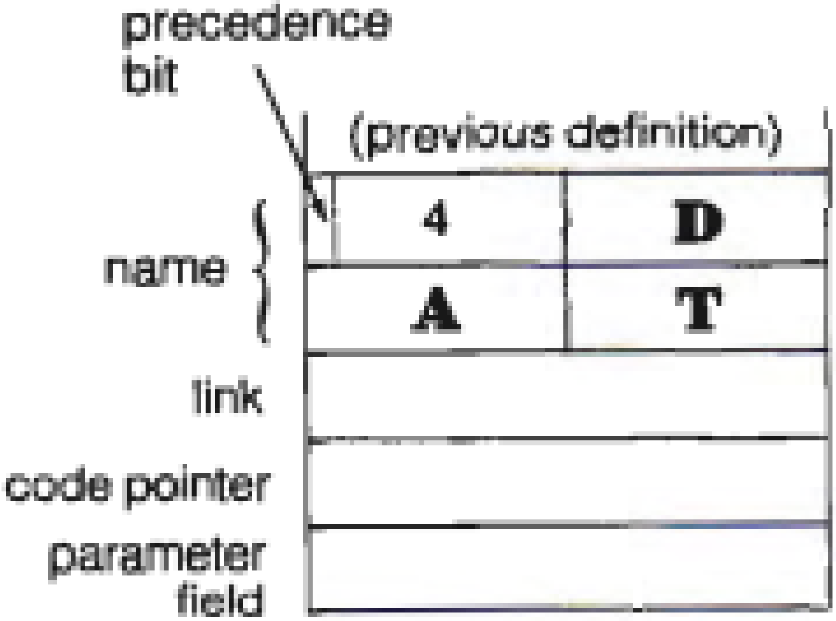
Paradigm 1: Procedural

- A procedural programming language is one that uses sets of functions and commands to complete actions. Many programming languages use the procedural programming paradigm, including BASIC, C and **Forth**.
- It is derived from imperative programming, based on the concept of the *procedure call*. **Procedures** (a type of routine or subroutine) simply contain a series of computational steps to be carried out.
- The following are the key functions of Procedural Programming Paradigm
  1. Pre-Defined Functions
  2. Modularity
  3. Top-Down Approach
  4. Parameter Passing
- Data in procedural programming is often declared as global variables, accessible throughout the program
- This can lead to potential issues such as naming conflicts and difficulty in maintaining large programs
- Procedural programming focuses on designing and implementing algorithms to solve problems.

Language for Paradigm 1: Forth

- **Forth** is a procedural, stack-oriented programming language and interactive development environment designed by Charles H. "Chuck" Moore and first used by other programmers in 1970.
- Forth, the computer language, was created for programming embedded and real-time applications. Today, it is available for developing applications on Windows, DOS, and variants of Unix that include macOS. Additionally, commercial-grade Forth cross compilers generate highly optimized code that runs on a variety of microprocessors and micro controllers, and prove themselves very capable in custom-hardware environments.
- **Forth** is a heavily stack-oriented self-compiling procedural programming language that is only slightly more abstract than assembly.
- Forth typically combines a compiler with an integrated command shell, where the user interacts via subroutines called words. Words can be defined, tested, redefined, and debugged without recompiling or restarting the whole program.
- Each word and its definition are entered into FORTH's "dictionary." The dictionary already contained many words when you started, but your own words are now in the dictionary as well. When you define a new word, FORTH translates your definition into dictionary form and writes the entry in the dictionary. This process is called "compiling. "
- In some versions of FORTH, including polyFORTH, the compiler does not copy the entire name of the definition into the dictionary–only the first three characters and the number of characters. For example, in polyFORTH, the text interpreter cannot distinguish between STAR and STAG because both words are four characters in length and both begin S-T-A.
- Vectored Execution : The idea of vectored execution is really quite simple. Instead of executing a definition directly we can execute it indirectly by keeping its address in a variable, then executing the contents of the variable.
- The advantage is that we can change the pointer later, so that a single word can be made to perform different things at different times.
- Using the variable DATE as an example, here's how these components are arranged within each dictionary entry in systems that have a

three-character-maximum name field. In this diagram, each horizontal line represents one cell in the dictionary:



In the above C example, there is a function named "calculate area" which computes the area of circle with the radius as the input

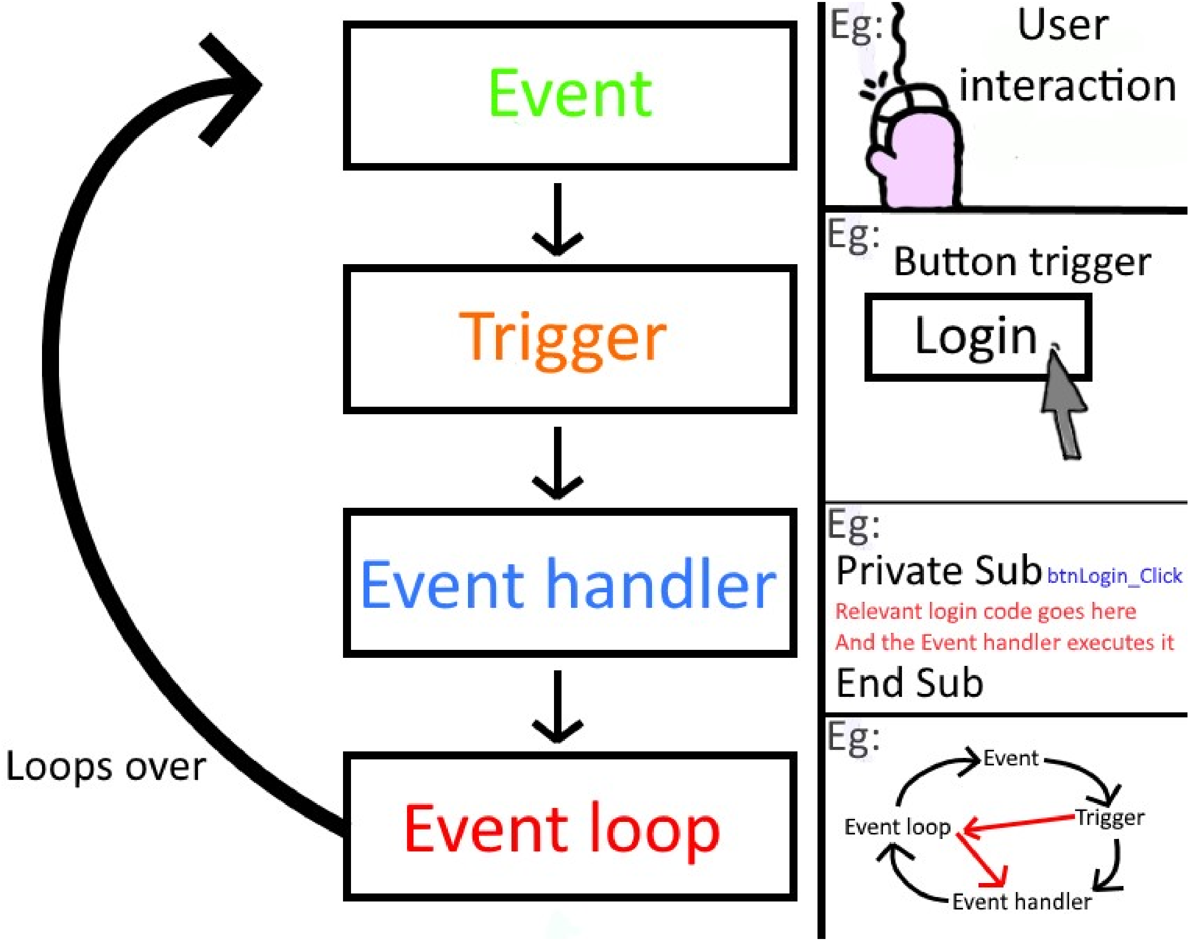
Paradigm 2: Event-Driven

- Event Driven Programming is a programming approach that enables software to respond to events originating from external sources, such as user input or system changes.
- It allows the creation of dynamic applications where the flow of control is determined by the sequence of events, rather than a predetermined order of execution.
- The primary goal of Event Driven Programming is to make the software more responsive to user actions and to simplify the development process

by providing a clear separation between event handling and other aspects of the software design.

- Event-driven programming often involves asynchronous execution, where the program doesn't wait for events to complete before moving on to the next task. Asynchronous handling allows the program to remain responsive and handle multiple events simultaneously.

- Event-driven programming is commonly used in GUI programming, where user interactions with buttons, menus, and other interface elements trigger events. Examples include handling button clicks, responding to mouse movements, or processing keyboard input.



Language for Paradigm 2: Angular

- **AngularJS** is a Javascript open-source front-end structural framework that is mainly used to develop single-page web applications.
- AngularJS was originally developed in 2009 by Misko Hevery and Adam Abrons. It is now maintained by Google.
- Event-Driven Architecture (EDA) is a software design pattern that revolves around the production, detection, and consumption of events. In this pattern, components within the system communicate with each other by exchanging events.
- An event is a lightweight message that represents a significant occurrence or a state change within the system. Events are typically processed asynchronously, allowing the system to continue executing other tasks in parallel.
- AngularJS is a tool-set for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs.

Analysis

Procedural Programming Paradigm

- Developers can use procedural coding languages for most general programming uses. Procedures in their respective languages are easy to understand. There are also options to create a variety of commands for a variety of conditions.
- Procedural programming values sequential functions and operations over actual data. This means it may not be the best programming language for projects that involve data-sensitive materials.
- The focus of procedural programming is to break down a programming task into a collection of variables, data structures, and subroutines. In other programming paradigms, it is required to break down a programming task into objects that expose behavior

Event-driven Programming Paradigm

- Event Driven Programming allows applications to effectively respond to user input, resulting in a more dynamic and user-friendly experience. Asynchronous event handling enables applications to execute multiple tasks concurrently.
- The asynchronous nature of event-driven applications can increase the software's complexity.
- Modularity and Reusability are two key features of Event Driven Programming that make it simple to construct, maintain, and enhance applications. In this paradigm, event handlers and other components are separated, allowing developers to compose their applications using well-defined modules.

Comparison

1. **Procedural Programming** determines its next set of statements by explicit instructions or procedures. **Event-driven programming** takes a different approach as it listens to the events that is influenced by user input or messages from the system.
2. In **Procedural Programming** the program executes linearly with a sequence of steps. In **Event-driven Programming**, the commands are not in any order,but depended upon the events that are triggered.
3. Corresponding to the previous point, we could see that the Forth Language uses "words" (procedures) that define next upcoming instructions. Angular JS depends on "event-handlers" and listens to these events and then call the respective function according to the event.
4. The code is more straight-forward and easy to understand in **Procedural Programming Language**. The code is more complex in **Event-driven Programming** as the flow and the logic of the program is driven by events.
5. Concurrency is typically managed in **Procedural Programming** using threads or processes if needed. It can be explicit and requires careful synchronization. In **Event-Driven Programming**, concurrency is inherent in the model. Multiple events can be processed concurrently without explicit management of threads.

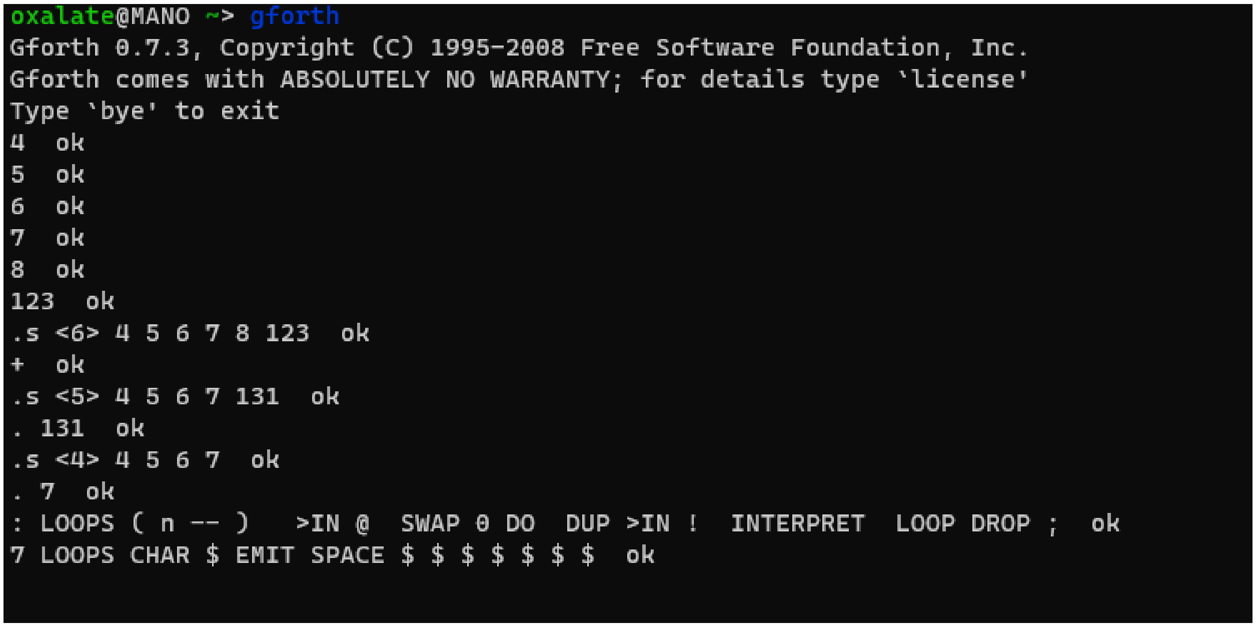
Comparison with other languages that follow the same paradigm

- Procedural Programming Paradigm is implemented in **C** in the form of functions. Data is passed along these functions as arguments and each function works .
- **Pascal** is also a procedural programming language, but instead of only using functions, it also has a separate keyword called a "procedure". Procedure is the same as a function but the only difference is that there is no return type.
- **Fortran** has a similar syntax to Pascal and has SUBROUTINE and Functions.
- Event-driven Programming Paradigm is followed by **Python** using the "asyncio" library. The asyncio module was added into Python in version 3.4, and it is available on all the later versions of Python. Asyncio module provides a very good infrastructure for writing concurrent code as single-threaded using the coroutines in the program.
- In **Java** event-driven programming, there are two primary components: the event listener and the event source. The event source is the object that generates the event, while the event listener is the object that receives and processes the event.

Examples

Procedural Paradigm :

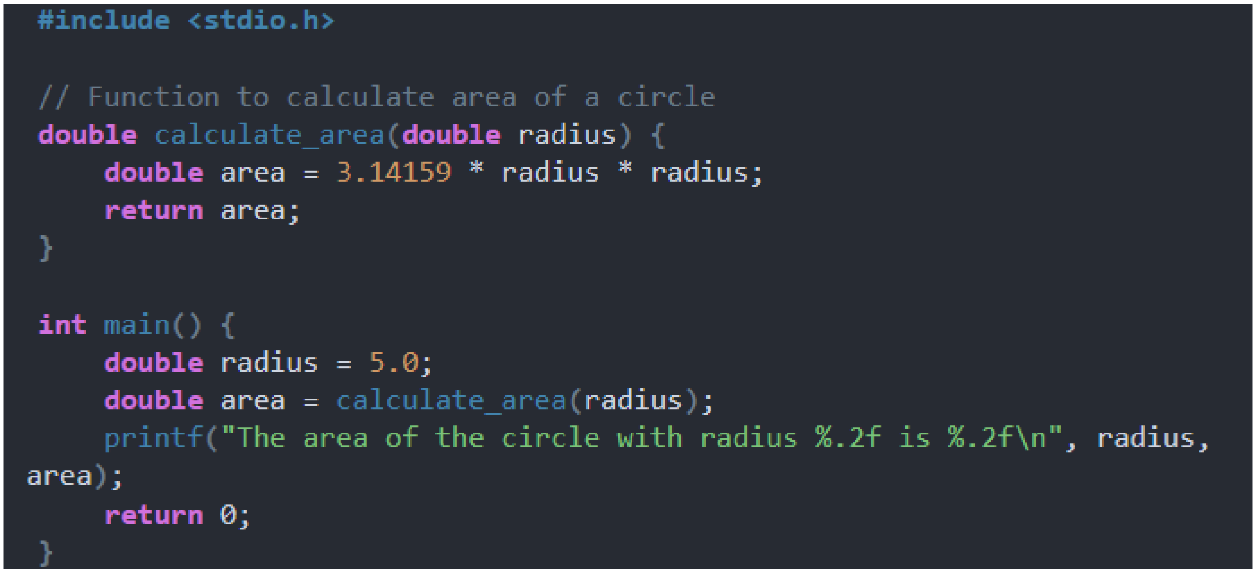
- Forth:



```
oxalate@MANO ~> gforth
Gforth 0.7.3, Copyright (C) 1995-2008 Free Software Foundation, Inc.
Gforth comes with ABSOLUTELY NO WARRANTY; for details type `license'
Type `bye' to exit
4  ok
5  ok
6  ok
7  ok
8  ok
123  ok
.s <6> 4 5 6 7 8 123  ok
+  ok
.s <5> 4 5 6 7 131  ok
. 131  ok
.s <4> 4 5 6 7  ok
. 7  ok
: LOOPS ( n -- )   >IN @  SWAP 0 DO  DUP >IN !  INTERPRET  LOOP DROP ;  ok
7 LOOPS CHAR $ EMIT SPACE $ $ $ $ $ $ $  ok
```

In this Forth example, all the values are stored inside a stack and "postfix" when an operation or a pre-defined word is entered, the top value of the stack is often taken and used. Loops is an user defined word which can print any string n number of times.

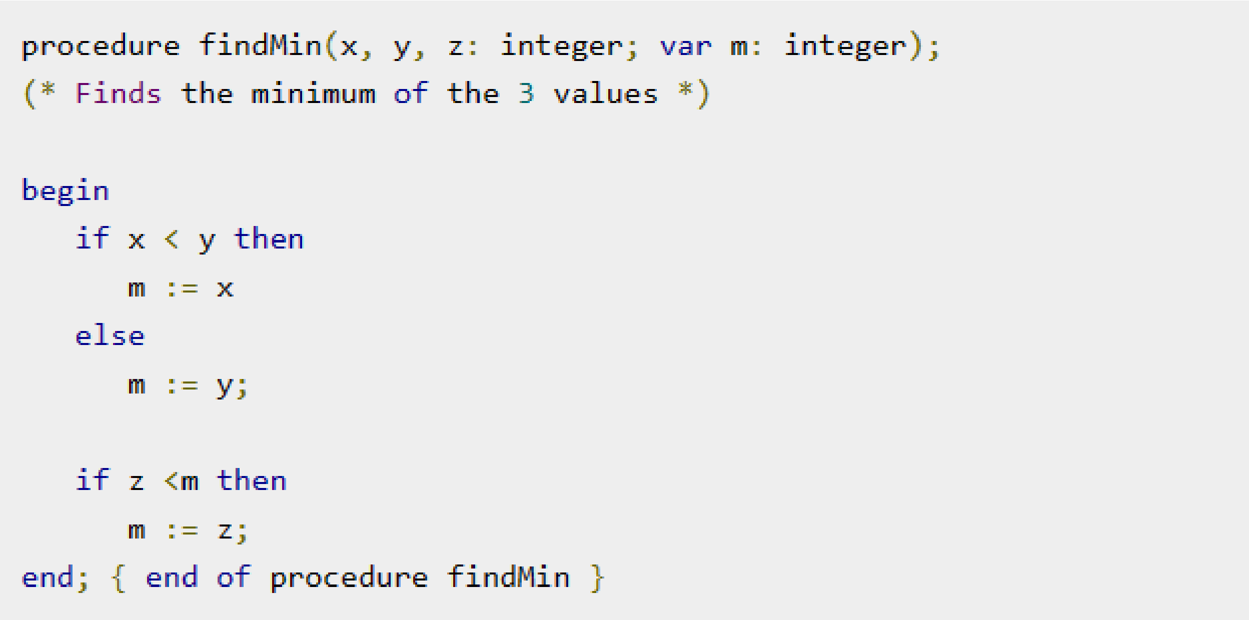
- C:



```c
#include <stdio.h>

// Function to calculate area of a circle
double calculate_area(double radius) {
    double area = 3.14159 * radius * radius;
    return area;
}

int main() {
    double radius = 5.0;
    double area = calculate_area(radius);
    printf("The area of the circle with radius %.2f is %.2f\n", radius,
area);
    return 0;
}
```

In the above C example, there is a function named "calculate area" which computes the area of circle with the radius as the input.

- Pascal:

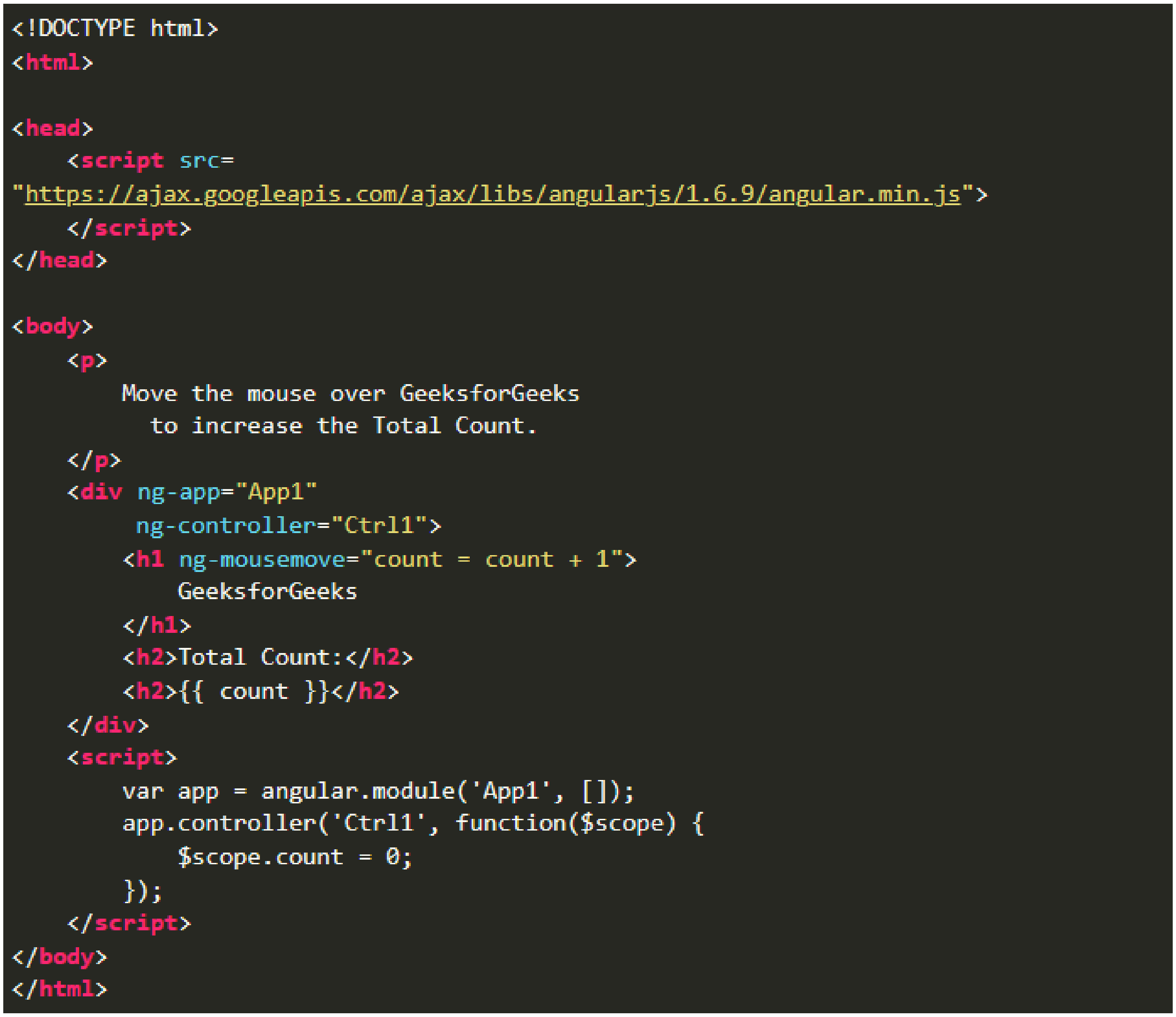


```pascal
procedure findMin(x, y, z: integer; var m: integer);
(* Finds the minimum of the 3 values *)

begin
   if x < y then
      m := x
   else
      m := y;

   if z <m then
      m := z;
end; { end of procedure findMin }
```

In the Pascal program given above, findMin is a procedure that finds minimum of the three numbers, x,y or z. The program follows a sequential flow of execution.

Event-driven Paradigm :

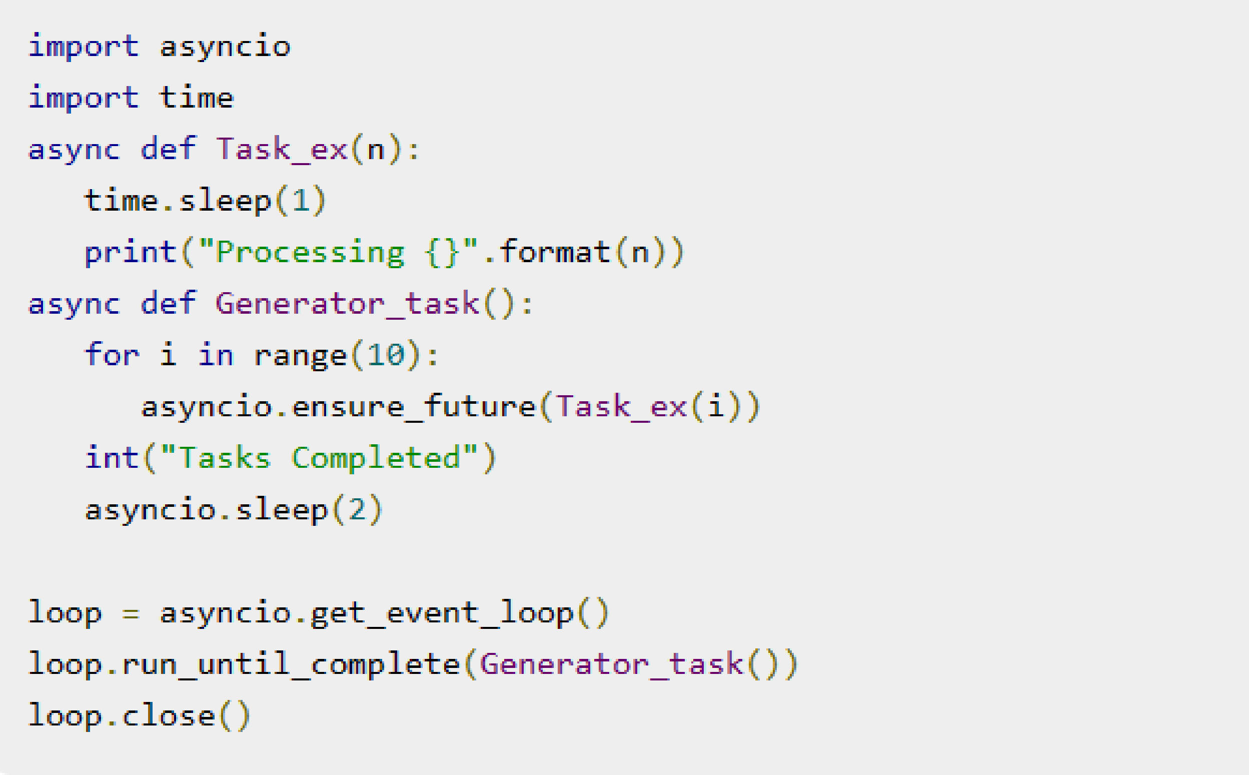
- AngularJS:



```
<!DOCTYPE html>
<html>

<head>
    <script src=
"https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js">
    </script>
</head>

<body>
    <p>
        Move the mouse over GeeksforGeeks
          to increase the Total Count.
    </p>
    <div ng-app="App1"
         ng-controller="Ctrl1">
        <h1 ng-mousemove="count = count + 1">
            GeeksforGeeks
        </h1>
        <h2>Total Count:</h2>
        <h2>{{ count }}</h2>
    </div>
    <script>
        var app = angular.module('App1', []);
        app.controller('Ctrl1', function($scope) {
            $scope.count = 0;
        });
    </script>
</body>
</html>
```

In this AngularJS example, the execution of the event triggers the variable count to increment by 1. The event is hovering the mouse over the text.

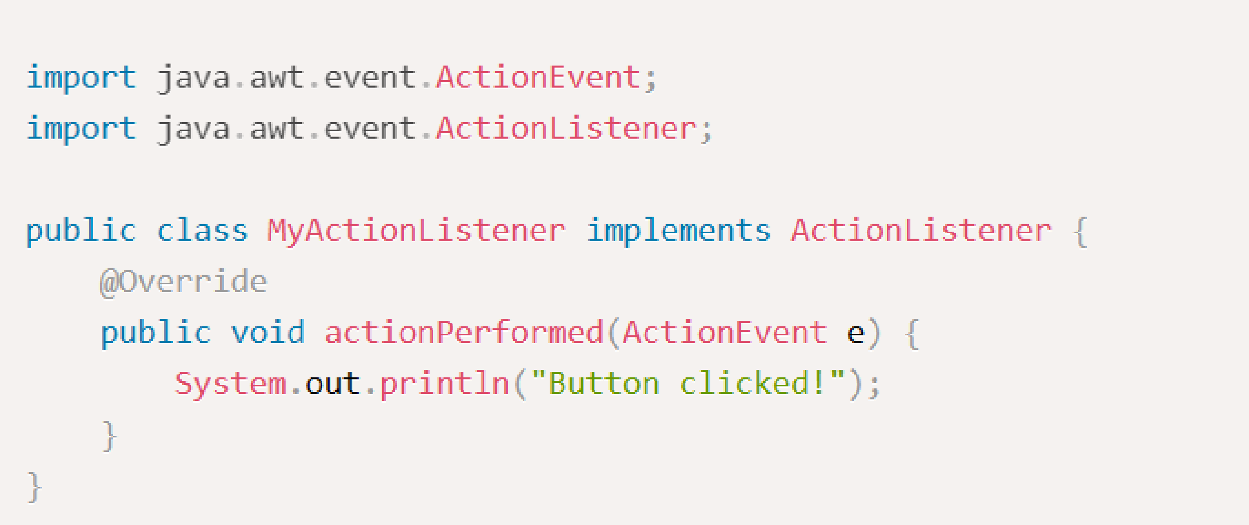
- Python:



```python
import asyncio
import time
async def Task_ex(n):
   time.sleep(1)
   print("Processing {}".format(n))
async def Generator_task():
   for i in range(10):
      asyncio.ensure_future(Task_ex(i))
   int("Tasks Completed")
   asyncio.sleep(2)

loop = asyncio.get_event_loop()
loop.run_until_complete(Generator_task())
loop.close()
```

In the above python example, the asyncio module is used along with the time module. The asyncio module ensures that multiple asynchronous tasks are completed parallely. "Tasks" subclass of Asyncio module is responsible for execution of coroutines within an event loop in parallel manner.

- Java:



```java
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MyActionListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        System.out.println("Button clicked!");
    }
}
```

This is an example Java program that uses Java AWT(Abstract Window Toolkit) which is an API to develop Graphical User Interface (GUI) or windows-based applications in Java. Here we would have to implement a method "addActionListener", which would as the name suggests, add a response to the event, i.e., clicking on a button.

Real-World Usage

FORTH

- Rockwell produced single-chip microcomputers with resident Forth kernels: the R65F11 and R65F12. ASYST was a Forth expansion for measuring and controlling on PCs.
- Space Shuttle Scientific Instrumentation Interface: The following material is taken from the paper, "Forth in Space: Interfacing SSBUV, a Scientific Instrument, to the Space Shuttle," written by Robert T. Caffrey for the June, 1992, Rochester Forth Conference. It describes the development of the Forth-based Small Payload Accommodations Interface Module (SPAIM), which interfaces the Shuttle Solar Backscatter Ultraviolet (SSBUV) instrument to the Space Shuttle's avionic systems.

AngularJS

- AngularJS framework is employed by Netflix for developing the user interface of its streaming platform. Leveraging two-way data binding, dependency injection, and modular design, AngularJS enhances the efficiency of Netflix's web applications, ensuring seamless and responsive user experiences.

- Since Gmail was developed as a single-page emailing application in 2004, it served as the perfect use case for integrating AngularJS.

  Every time a user opens Google's mail service, the web app loads one single HTML page whilst dynamically updating information based on new messages or notifications.

- PayPal offers a safe wallet mechanism for both senders and recipients to transfer cash. Both must create an account with PayPal and give information about credit, debit, or bank account that will be used if there is insufficient money in the PayPal wallet to fulfill a transaction. The account may only be accessed using an email address and a password.
  Front-end: For the front-end, it leverages React, Node. js/Kraken, and AngularJS framework with JavaScript programming. PayPal supports payment options on the website, allowing customers to have a more customized and efficient checkout experience. PayPal was able to determine the best financing sources and buttons to offer purchasers with the help of customized integration that sent query and script parameters in JavaScript.
  Backend: The backend tech stack is the server-side of software development that includes Java with the Spring Boot framework and REST, with Go and Scala+Akka as alternatives. PayPal has opted to utilize JavaScript for web applications from the browser to the backend server. PayPal's backend tech stack uses the mix of Squbs, a reactive framework based on Akka, Visual Basic.NET (VB.NET), and MongoDB.

Advantages of Languages for each Paradigm

FORTH

- **Efficiency**:

  FORTH is known for its efficiency and performance, especially in embedded systems and resource-constrained environments. Its minimalistic design and stack-based execution contribute to efficient use of memory and execution speed.

- **Interactivity**:

  FORTH provides an interactive programming environment, allowing developers to directly interact with the system through a command-line interface. This interactivity facilitates rapid development, testing, and debugging.

- **Extensibility**:

  FORTH is highly extensible, allowing developers to add new commands and functionalities easily. This extensibility promotes code reuse and the creation of domain-specific languages tailored to specific tasks.

- **Portability**:

  FORTH's design philosophy focuses on portability, making it well-suited for a wide range of hardware platforms. The minimalistic nature of FORTH implementations allows for easy adaptation to diverse environments.

- **Embedded Systems**:

  FORTH is commonly used in embedded systems due to its minimal footprint, efficiency, and suitability for low-level programming tasks. Its direct interaction with hardware makes it a preferred choice for embedded applications.

AngularJS

- **Modularity**:

  AngularJS promotes a modular architecture, allowing developers to break down applications into smaller, manageable components. This modularity enhances code organization, reusability, and maintainability.

- **Two-Way Data Binding**:

  AngularJS features two-way data binding, enabling automatic synchronization between the model and the view. Changes in the user interface (view) or data model are reflected in real-time, reducing the need for manual updates.

- **Dependency Injection**:

  AngularJS uses dependency injection to manage and inject dependencies into components. This promotes loose coupling, testability, and easier maintenance by allowing components to be easily replaced or updated.

- **Declarative UI**:

  The framework supports a declarative approach to building user interfaces, where developers specify the desired outcome rather than the step-by-step process. This makes the code more readable, expressive, and easier to understand.

- **Event-Driven Architecture**:

  AngularJS is built on an event-driven architecture, where user interactions (events) trigger responses through event handlers. This approach enables the creation of dynamic, responsive user interfaces that react to user input in real-time.

- **Community and Ecosystem**:

  AngularJS benefits from a large and active community, providing a wealth of resources, libraries, and third-party modules. The ecosystem around AngularJS supports rapid development through a rich set of tools and extensions.

- **Cross-Browser Compatibility**:

  AngularJS abstracts away many browser-specific inconsistencies, ensuring better cross-browser compatibility. Developers can focus on building features rather than addressing browser compatibility issues.

- **Testing Support**:

  AngularJS is designed with testability in mind, providing support for unit testing and end-to-end testing. This facilitates the development of robust and reliable applications with a focus on quality assurance.

Disadvantages of Languages for each Paradigm

FORTH

- **Learning Curve**:

  FORTH has a unique and unconventional syntax, which can be challenging for developers who are accustomed to more mainstream procedural languages. This may result in a steeper learning curve.

- **Limited Standardization**:

  FORTH lacks a standardized specification, leading to variations in implementations. This lack of standardization can make it difficult to write portable code that works seamlessly across different FORTH systems.

- **Debugging Challenges**:

  Debugging in FORTH can be challenging due to its stack-based nature. Traditional debugging tools and techniques used in other procedural languages may not be as effective, making it harder to identify and resolve issues.

- **Limited Library Support**:

  FORTH has a relatively small standard library compared to more mainstream languages. This may require developers to implement many functionalities from scratch, leading to increased development time.

- **Portability Issues**:

  FORTH programs may not be as portable across different platforms and systems due to the lack of standardized features. Code that works on one FORTH system might need modifications to run on another.

AngularJS

- **Performance Overhead**:

  AngularJS introduces a layer of abstraction for data binding and event handling, which can lead to some performance overhead. In scenarios with large and complex applications, this overhead may impact the overall performance.

- **Steep Learning Curve**:

AngularJS, being a comprehensive framework, has a steep learning curve, especially for beginners. Developers need to grasp various concepts like directives, controllers, and services, which may take time to master.

- **Dependency on JavaScript**:

  AngularJS relies heavily on JavaScript, and applications built with it may not function well or at all when JavaScript is disabled in a user's browser. This dependency can affect accessibility and SEO (Search Engine Optimization).

- **Versioning and Migration**:

  AngularJS has undergone significant changes with the release of Angular (2 and later). Migrating from AngularJS to the latest versions of Angular can be a complex process, and there's a lack of backward compatibility.

- **Complexity for Small Projects**:

  For small projects, the extensive features and structure provided by AngularJS may introduce unnecessary complexity. Using AngularJS for simple applications may be overkill and result in additional development effort.

Limitations in each Paradigm

Procedural Programming Paradigm :

- **Limited support for abstraction**: Although procedural programming facilitates abstraction through functions and procedures, it falls short in delivering more advanced abstraction mechanisms offered by other paradigms, like object-oriented and functional programming.
- **Difficulty handling large-scale projects**: As projects grow in size and complexity, managing the separation of concerns and data sharing between functions and procedures can become challenging. This makes procedural programming less suited for very large-scale or complex projects with many interacting components.
- **Lack of support for inheritance and polymorphism**: Procedural programming does not provide support for inheritance and polymorphism, which are essential features in the object-oriented paradigm. This can limit code reusability and make it more challenging to create flexible and extendable code structures.

- **Global data and side effects**: Procedural programming often makes use of global data and variables that can lead to side effects, making it more difficult to reason about the code's behaviour and maintain its integrity. This contrasts with functional programming, which emphasizes immutability and the avoidance of side effects.
- **Concurrency challenges**: Procedural programming's reliance on mutable data structures and state can make it more challenging to handle concurrent and parallel execution compared to functional programming, which promotes immutability and statelessness, thus simplifying concurrency management.

Event-Driven Programming Paradigm :

- **Complexity**: The asynchronous nature of event-driven applications can increase the software's complexity. Ensuring correct synchronization, managing race conditions, and addressing deadlock scenarios may necessitate extensive effort and precision in coding.
- **Debugging Difficulties**: Debugging event-driven applications can prove challenging, especially when dealing with concurrency, as the order of event execution is not predetermined and may vary during runtime. This unpredictability can complicate the process of identifying and resolving issues.
- **Event Handling Overhead**: The execution of event handlers and management of events demand additional system resources. Furthermore, the event loop and event queue require constant monitoring, potentially impacting performance.
- **Steep Learning Curve**: Developers who are unfamiliar with Event Driven Programming may experience a steep learning curve, particularly when grappling with complex concurrency behaviours and synchronisation.
- **Dependencies on External Libraries**: In certain programming languages and environments, Event Driven Programming might rely on external libraries for managing events and handling asynchronous tasks. This dependence on external code may complicate deployment and maintenance.

Challenges Faced

The main challenge during the exploration of programming paradigms is :

- Forth is a very niche programming language and is used in few areas such as embedded systems, spreadsheets and in databases. So it was hard to find different example programs.
- Forth may pose challenges in terms of readability and learning curve due to its unconventional syntax and stack-based nature. This could be a limitation for developers more accustomed to traditional procedural languages.
- Both the programming paradigms and the languages (AngularJS and Forth) are implemented in different areas which made it difficult to find consistent examples.

Conclusion

- **Procedural Programming Languages** are used for tasks that need to be completed step by step. Procedural Programming Paradigm simply contains instructions or procedures that are executed step by step .
- Forth is a procedural programming language that is stack based and is simple to understand.
- Forth's stack-based execution model allows for efficient resource utilization, making it well-suited for embedded systems and real-time applications where performance is crucial.
- Forth supports an interactive development process through its interpreter. This feature enables developers to test and modify code incrementally, promoting an agile and exploratory approach.
- While **Event-driven Programming Languages** are usually more complex, it is used in almost every website that we come across.
- Angular JS is a Javascript framework that can be used to build single page applications and web based mobile applications that have user interactivity.
- AngularJS excels in building dynamic and interactive user interfaces. Features like two-way data binding and declarative templates contribute to creating engaging web applications.

- AngularJS promotes scalable and maintainable code through concepts like dependency injection, modularization, and separation of concerns. This contributes to the framework's suitability for large and complex applications.
- AngularJS efficiently handles asynchronous events, contributing to a smooth user experience by managing events such as user interactions and data updates.
- AngularJS benefits from a supportive and vibrant community. This community actively contributes to the framework's development and provides resources.

References

- https://www.forth.com/forth/
- https://users.ece.cmu.edu/ koopman/forth/hopl.html
- https://www.codementor.io/@joshuaudensi/getting-started-event-driven-programming
  cvb5mgbzh
- https://www.studysmarter.co.uk/explanations/computer-science/computer-programming/event-driven-programming/
- https://www.geeksforgeeks.org/angularjs-events/
- https://angularjs.org/