**Interactive Pascal’s Triangle (iPasT)**

**23 February 2018**

Fetalvero

Gueco

Naraval

Nolasco

**Preface**

**Readership**

This documentation is primarily aimed at mathematics lecturers utilizing the software in presenting Pascal’s Triangle. Students may find this documentation useful when attempting to create similar software. They may also use this paper as a guide for documenting related software. Readers are assumed to have basic knowledge in mathematics and computer science jargon.

**Rationale**

The main inspiration for creating the Interactive Pascal’s Triangle, iPasT for short, is to bring to life the notes of Dr. Jerico Bacani. A mere PowerPoint presentation is not enough to impart the gravity of the Pascal’s Triangle’s properties and phenomena. The Pascal’s triangle is in need of a different medium to be able to better portray its implications. In creating interactive software, users are enabled to discover the mysteries of the Pascal’s Triangle themselves, increasing their appreciation for the beauty of the mathematical patterns to be found in the triangle. Furthermore, an interactive system would better show the generality of the patterns and behavior of numbers in the triangle.

**Introduction**

Pascal’s triangle is a popular mathematical concept that is usually illustrated as a pyramid of numbers. The numbers in the triangle is generated using combinatorial mathematics. Usually utilized in binomial expansion, the triangle has numerous less known phenomena such as the hockey stick pattern which the software intends to illustrate. This project aims to educate as well as entertain end users with the magic of Pascal’s triangle.

The patterns to be found in Pascal’s triangle are usually illustrated in graphs and pictures. This type of portrayal albeit effective lacks life. To be able to grasp the emergence of a pattern, numerous pictures are needed to help the reader understand. Furthermore, most patterns only have few examples which keep the reader from understanding the generality of a certain concept.

In creating interactive software, the patterns and properties of the triangle will be animated thus increasing end user participation. Not only will the end users learn about the triangle’s phenomena they will also be able to witness how it happens in real time. Furthermore, they will be able to create their own examples as well as control how a certain pattern is simulated.

The project enables the user to explore the numerous properties and patterns to be found in Pascal’s triangle. Upon opening the application the user is prompted to enter the preferred number of rows to be presented. The intended user interface would include an animated Pascal’s triangle made of hexagons that houses each number in the triangle. Furthermore, the program would include other features such as highlighting the patterns to be found in the triangle, for example the Fibonacci sequence.

The software is strictly local. By localizing the system, internet access, which is vital in internet based software, is not necessary. This enables the end user to be able to utilize the system despite difficulties in network connection. The program performance will also become independent from the internet connection, thus ensuring constant efficiency and performance of the system. Although having internet based software would provide ease of accessibility, the programmers are limited in their resources. An online version of the system will be included in further developments of the software.

Overall, the system should be able to provide the end users with a means of illustrating the properties and phenomena of the Pascal’s triangle in an interactive manner.

**Glossary**

**End User -** the targeted audience for our program

**User Interface –** the visual presentation of the program, i.e., how the program is perceived by the end users

**Performance –** the overall efficiency and execution speed of the program

**Three tier architecture –** System architecture that describes the software with having three components: the user, the interface, and the database

**Unified Modeling Language (UML) –** a set of 13 different diagram types that may be used to model software systems

**Activity Diagram –** type of UML that shows the activities involved in the program

**Sequence Diagram –** type of UML that shows the interactions between actors and the system and between system components

**Use Case Diagram –** type of UML that show the interactions between a system and its environment

**User Requirements Definition**

**End User Services**

The software enables the end user to simulate a Pascal’s Triangle with a single row up to twenty. Furthermore, the software should enable the user to manipulate certain parts of the triangle to be able to come up with the suggested patterns. Educational commentary describing the various patterns and phenomena being illustrated should also be included in the user interface.

**Nonfunctional System Requirements**

The software shall be available to the end user at all times. It must have a consistent performance, independent from the availability of internet connection. The software must only be available to the users of the unit in which the program installed.

**System architecture**

**Three-tier Architecture**

System Resources:

* Graphics
* Code
* Data

Pascal’s Triangle App

End User

**System Requirements Specification**

**Functional System Requirements**

The end user should be able to generate a Pascal’s Triangle. The generation of the triangle should include a prompt that implores the user to input the number of rows to be simulated (rows should be between one and twenty).

The software shall highlight specific patterns the end user wishes the program to illustrate. The end user must be able to select the pattern from a range of choices. Upon choosing, the pattern should be simulated as well as explained in the user interface.

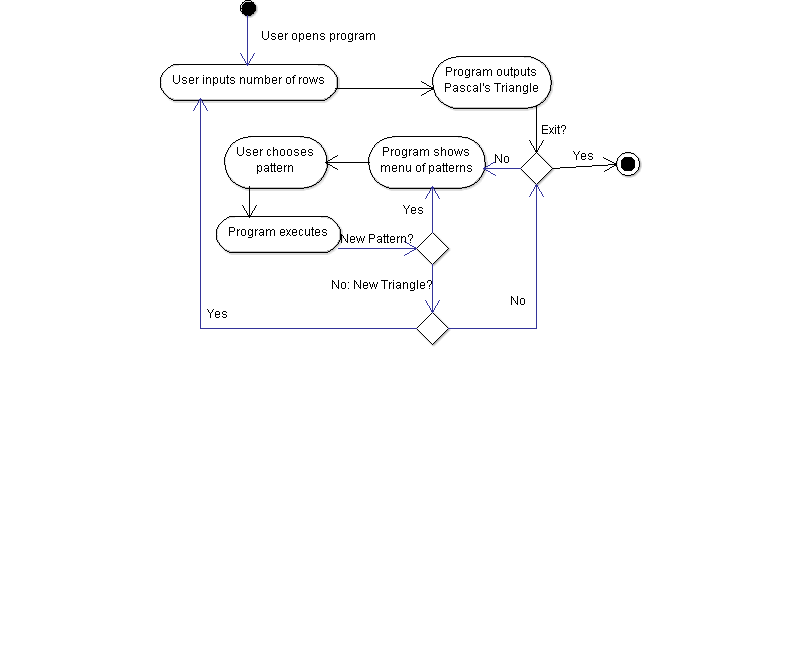
The program shall run at the respective operating system the end user operates on upon meeting the minimum system specifications which shall be covered at the introduction of the program.

**Nonfunctional System Requirements**

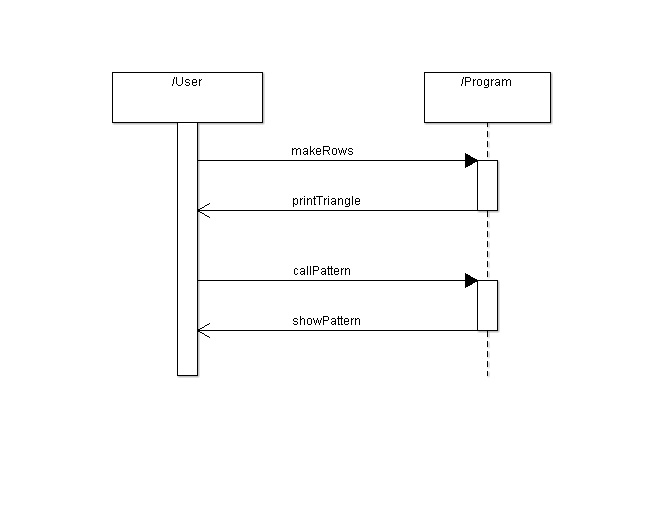
The software shall be available to the end user at all times. It must have a consistent performance, independent from the availability of internet connection. The software must only be available to the users of the unit in which the program installed.

The software should not hinder the learning process of its end users but be able to fully communicate the knowledge it was embedded with. This involves the reliability of the program upon use by its end users. The program should not have confusing and misinterpreted bugs that will bother its end users eventually.

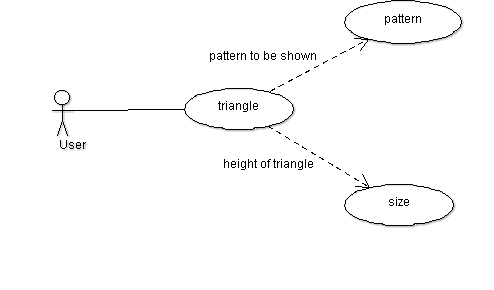
**System Models**



*Activity Diagram*



*Sequence Diagram*



*Use Case Diagram*

**System Evolution**

Further system evolution should include an internet based version. Hosting the software on the web would make it accessible to anyone who has a device and an internet connection. Global access would make the software along with the knowledge gained from it be open to critique also be open for people to learn from what it presents.

To future proof the system, further system evolution should be able to apply the concepts of modularity to further accommodate the new discoveries in relation to the Pascal’s Triangle. This should allow creative, fresh and new ways to the present the Triangle from its simple form.

Furthermore, the system should also grow to new mediums presentable through computer systems as well as unexplored ones. This may or may not include the recent Virtual Reality (VR) systems and unexplored such as touch-based, hearing based systems for the use of persons with disability. This growth will boost the audience by which the program for learning reaches out to.