CodeNect: Visual Programming Software for Learning Fundamentals of Programming

Brandon B. Lim-it, Jaykel O. Punay

January 7, 2021

Introduction

- Technology is constantly progressing and improving
- Programming is essential in the field of technology
- Programming is a discipline
- Programming is difficult
- Learning programming is more difficult

Introduction

According to Tsai, Yang, and Chang, one of the requirements for a programmer is to have expertise in technical skills that include multiple programming languages.

Introduction

Learning programming has indeed became easier and better through the use and aid of technology itself. Education integrates modern tools to increase the rate of knowledge acquisition and absorption (Raja and Nagasubramani, 2018).

Introduction

Commonly used software for programming have numerous features that are useful and engaging but for learning, the features become bloat and can result in user fatigue (Thompson, Hamilton, and Rust, 2005).

Statement of the Problem

The fundamental concepts of programming are essential basics that are necessary for programmers to master. Concepts such as:

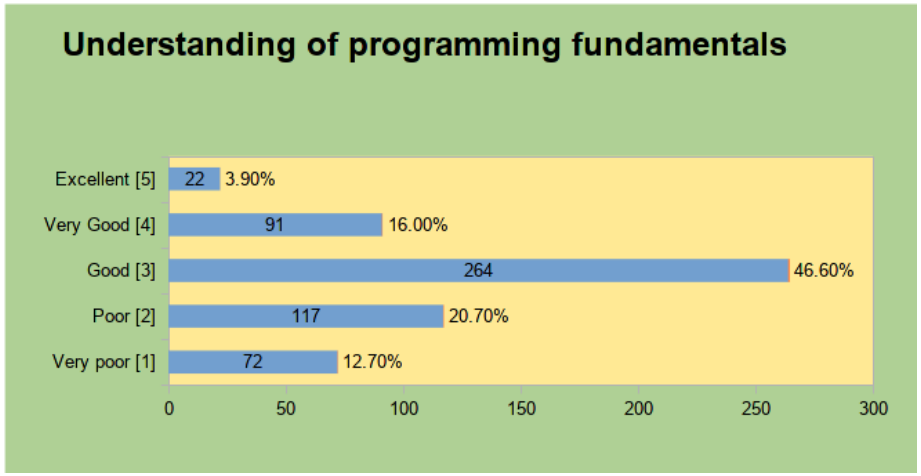
- Syntax and Semantics
- Data Types and Data Structures
- Logic and Conditionals
- Loops and Algorithm
- Memory

are key to easily understanding and getting better at programming as programming is a discipline (Prahofer, Hurnaus, Wirth, and Mossenbock, 2007).

Statement of the Problem

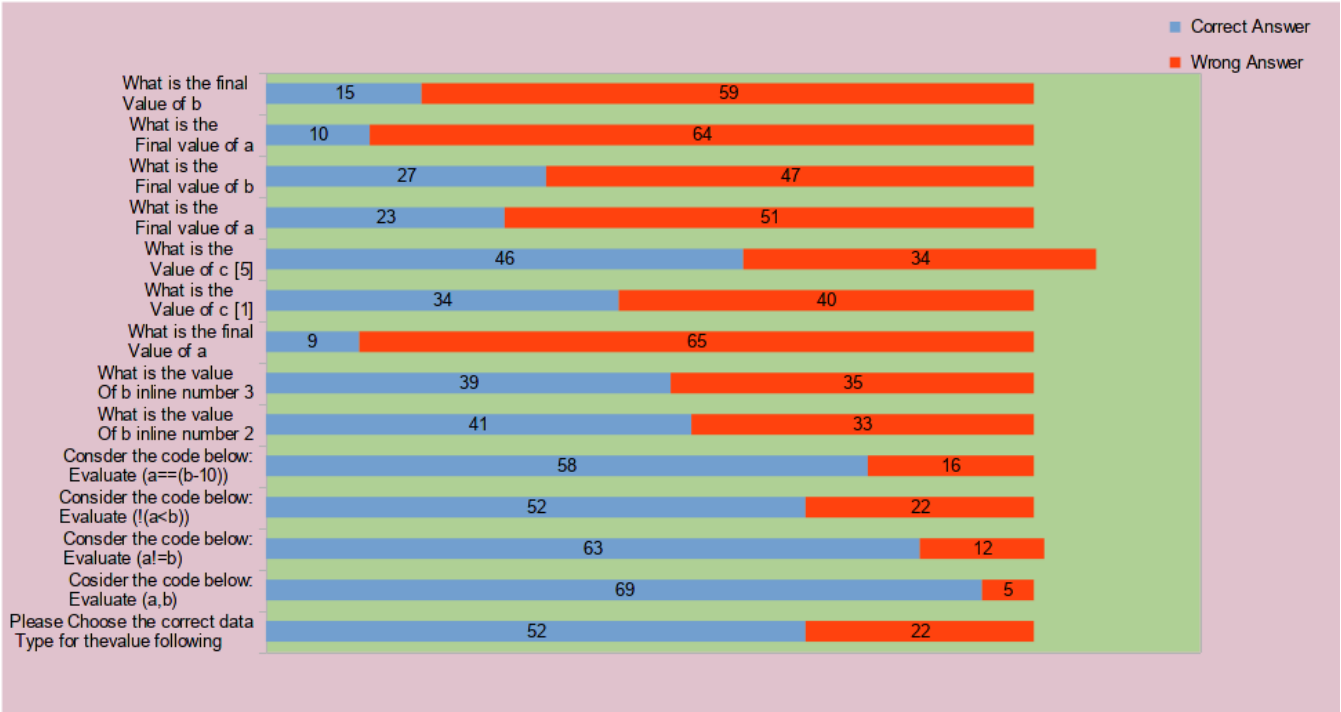
Programming is a skill which can be boring, intimidating, and unrelated to daily activities and experience. Programming education requires the assistance of technology itself through software in improving the quality of learning. The traditional method of pure lecture is nowadays complimented with the application of softwares. But most tools are not beginner-friendly and are cluttered with features that present confusion and steep learning curve in familiarity and mastery that diminish the learning experience (Tsukamoto et al., 2016).

Statement of the Problem



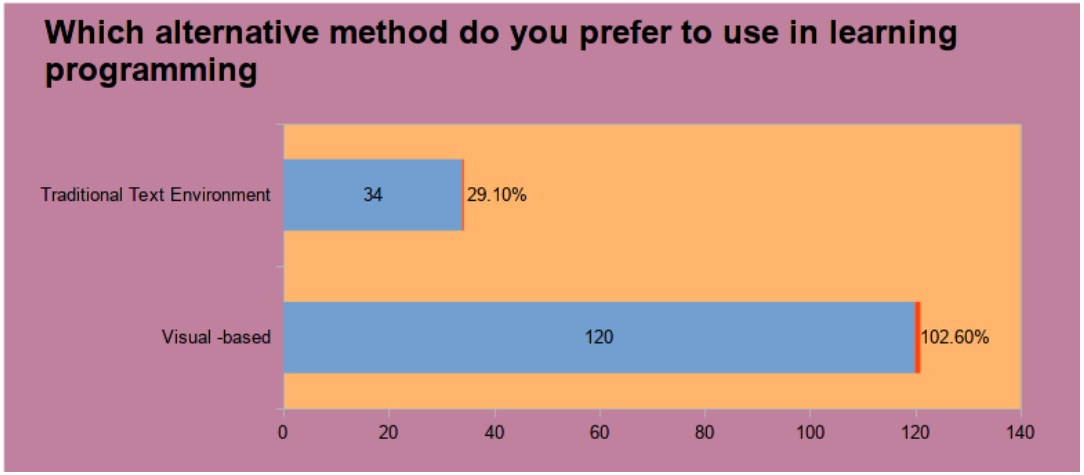
The assessment of the respondents under the courses with programming subjects shows that students are not familiar and not well versed on fundamental concepts and find it difficult to understand.

Statement of the Problem



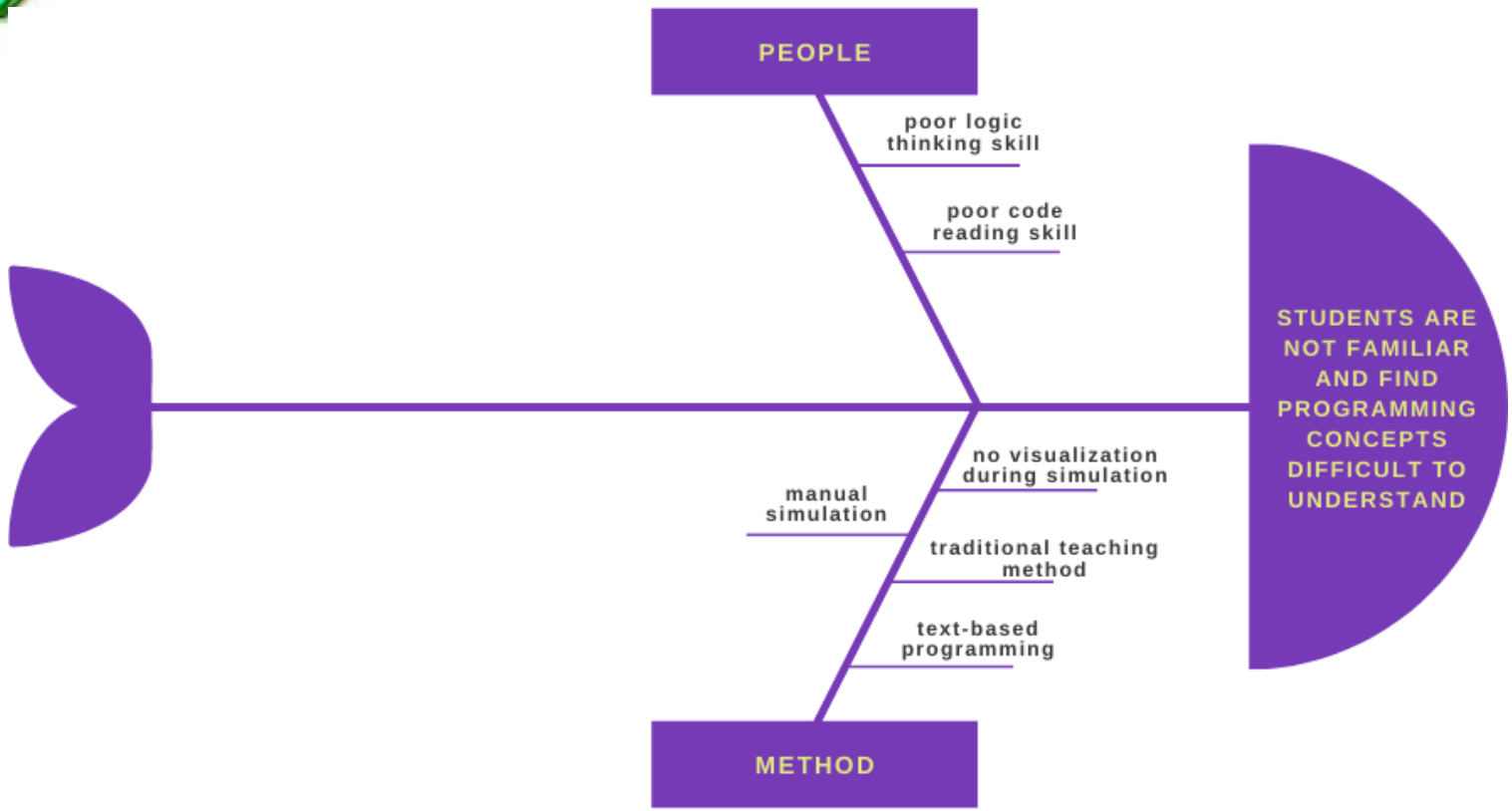
Basic concepts such as loops, memory management, and functions are easily understood individually, but combining them into a program has confused students. Respondents failed to correctly answer the assessment.

Statement of the Problem



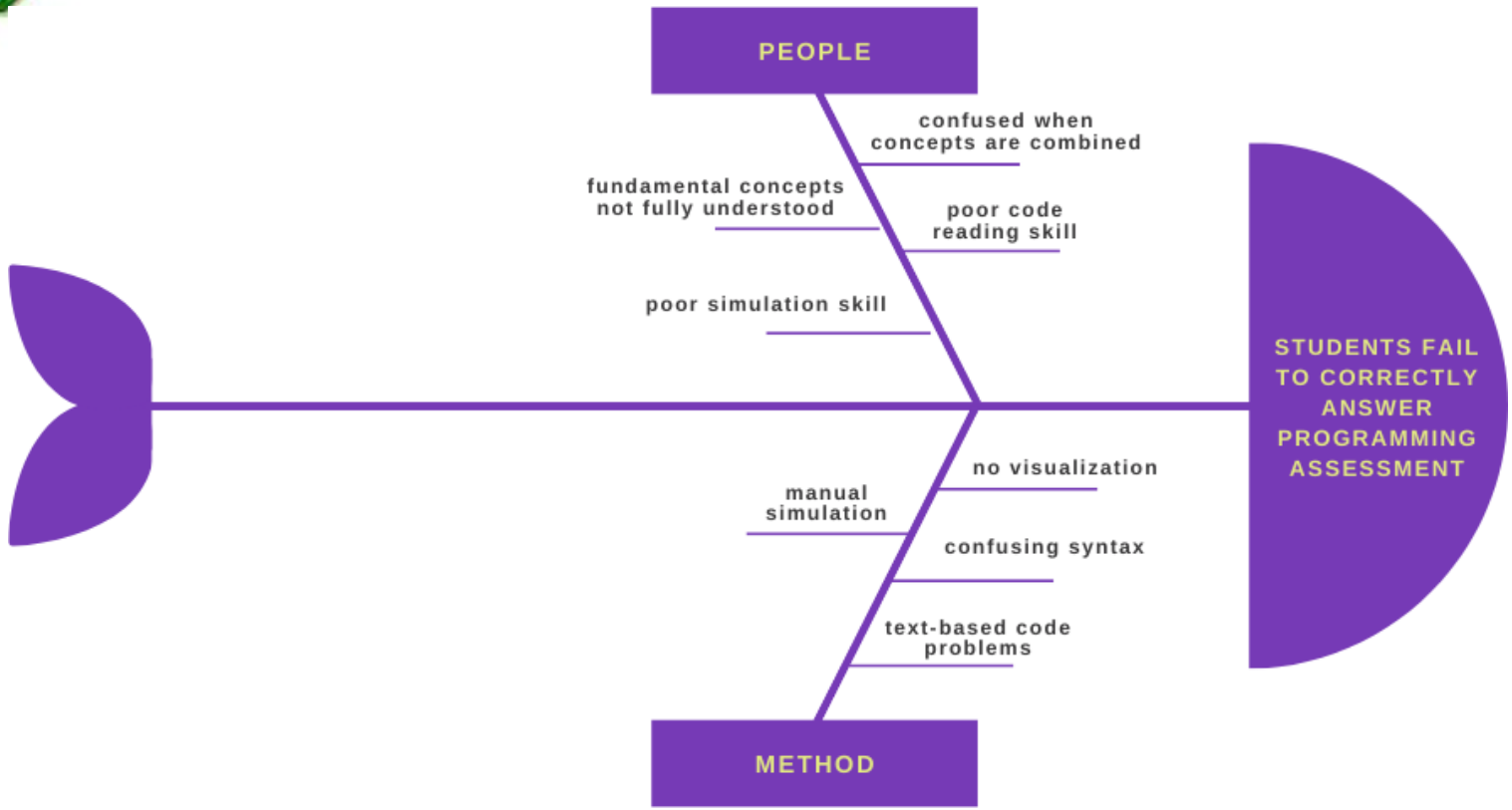
Survey shows that 76% of students use outdated text-based editors in their laboratory classes such as Notepad++, DevC++, and TurboC/C++ , while only 24% use professional and modern editors for programming. This traditional textbased editors are general tools and are not oriented for learning of beginners and thus not effective.

Ishikawa Diagrams



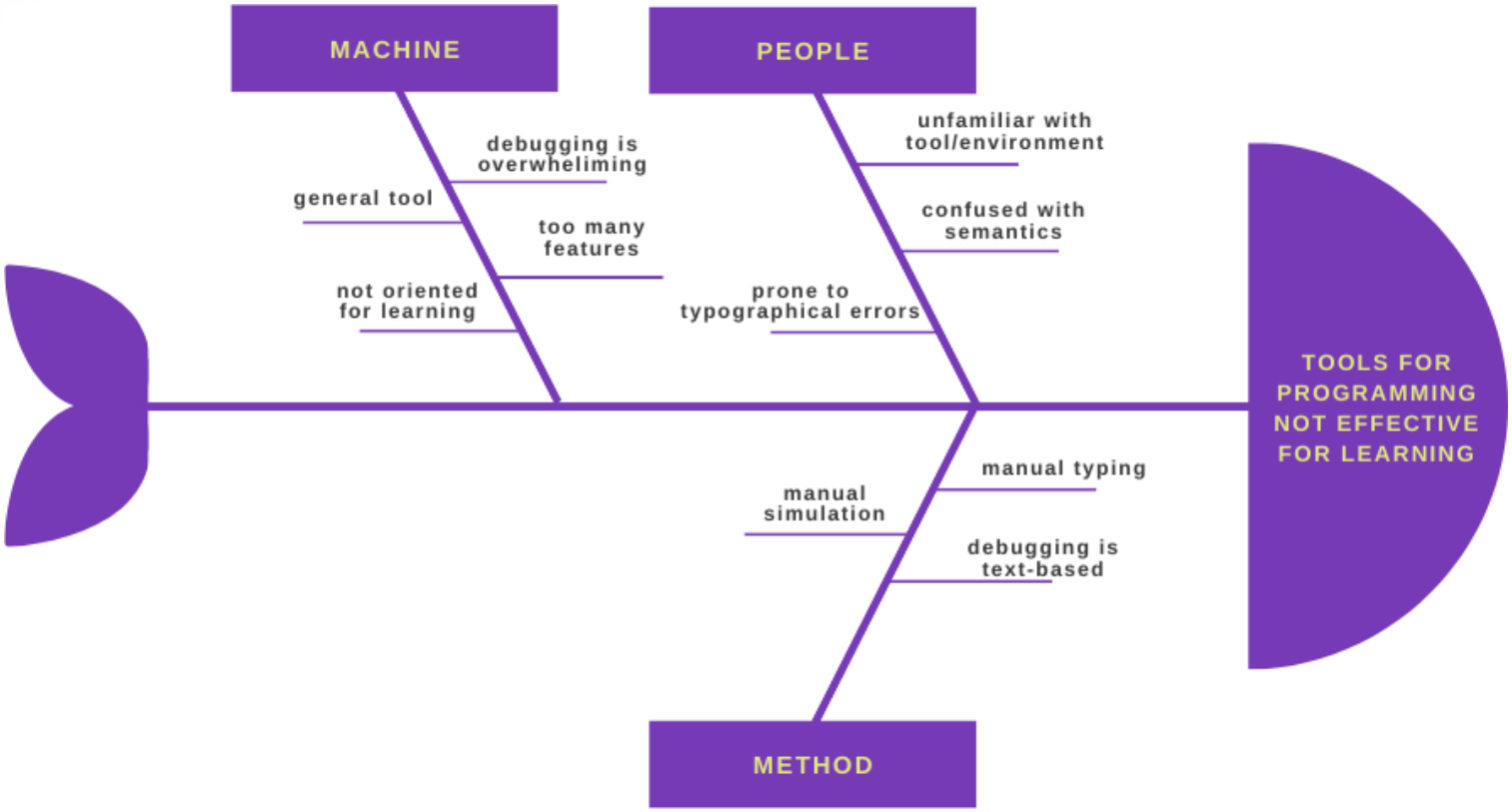
Ishikawa diagram of students not familiar and finding programming concepts difficult to understand

Ishikawa Diagrams



Ishikawa diagram of students failing to correctly answer programming assessment

Ishikawa Diagrams



Ishikawa diagram of the tools for programming not effective for learning

Objectives of the Study

The general objective of the study is to develop a CodeNect: Visual Programming Software that will help in learning the fundamentals of programming.

Specifically, this study seeks:

- Identify the concepts learners find difficult to understand through conducted survey.
- Analyze the problems through a Ishikawa/Fishbone Diagram.
- Design the system using the Use Case Diagrams.
- Test the usability, functionality of the software using Experience-based test design.
- Evaluate the acceptability of the software using the ISO/IEC/IEEE 29119-4.2015.

Objectives of the Study

- Develop the software with the following main features:
  - Visual Nodes Module
  - Filesystem Module
  - Input/Output Module
  - Debug Module
  - Simulation Module
  - Transpiler Module
  - Assessment Module

Significance of the Study

Students

The software will help in the education and improvement in the knowledge, skills, understanding, and expertise of the students and learners about programming. Thus, allowing them to compete and increasing the opportunities for their careers.

Teachers

The software will provide assistance for teachers and instructors to teach and demo programming concepts through visualization. This will aid in relieving workload, stress, and maximizing lessons each class time.

Significance of the Study

Educational Institutions

The software will benefit educational institutions like university for computer laboratory classes by providing a free software oriented for the purpose of learning

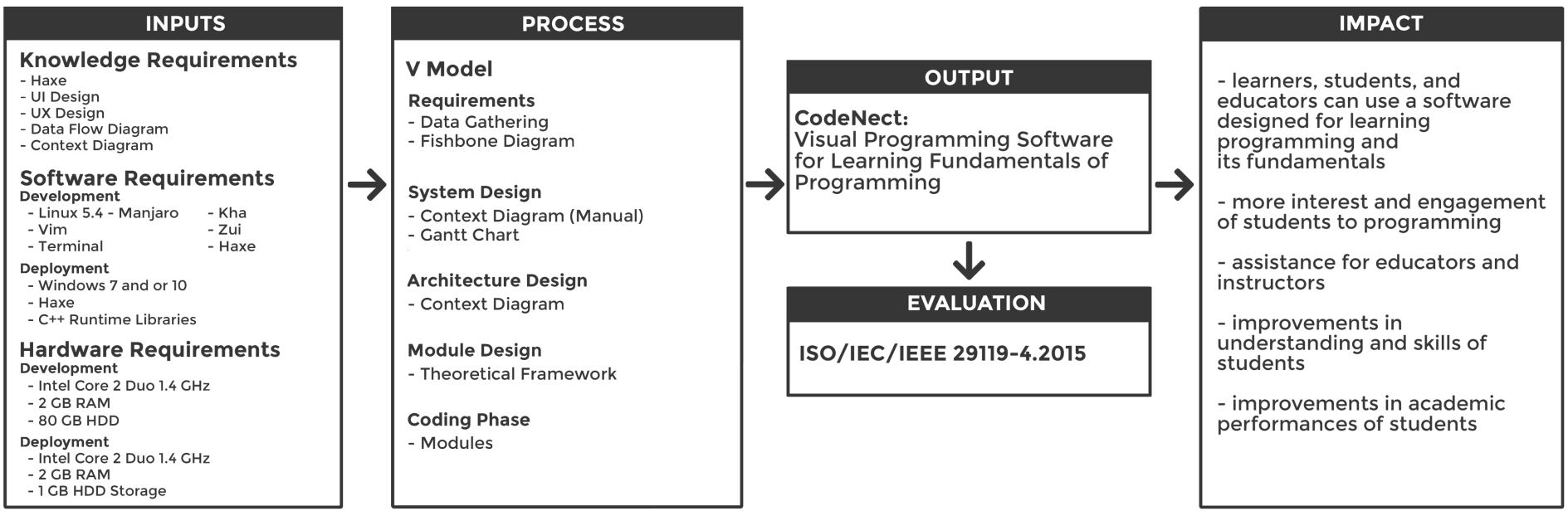
Developers

The software will provide learning experience for the developers and researchers in preparation for software development career.

Researchers

This study would serve as a guide and reference in the field of software development and education for future researchers.

Conceptual Framework of the Study



Scope and Limitations of the Study

Simplicity and Functionality

The software will prioritize simple and basic functionalities over numerous features for the purpose of learning and education.

Stand-alone Program

The software will have no account management and can be run without any hassle. The software works perfectly in offline mode.

Terminal-based

The software is limited to simulating text-based or command/terminal prompts as the priority is learning the fundamentals of programming.

Scope and Limitations of the Study

Visual Nodes Module

Nodes are graphical elements that serve as the building blocks of the software. Nodes can be used as a variable, logic, and conditionals

Filesystem Module

Serves as the interface between the software and the user's machine for handling files such as creation, modification, reading, and deletion.

Scope and Limitations of the Study

Input/Output Module

The module is responsible for processing and responding events and performing actions based on the event such as key press, mouse click, and mouse movement.

Debug Module

This module will linter and give feedback and indication to the user whenever there is an attempt to perform an action that is faulty in logic

Scope and Limitations of the Study

Simulation Module

The process of simulation involves the compiling, building, and running the visual code is executed by this module

Transpiler Module

This module transpiles the visual code made by the user into source code in target programming language

Scope and Limitations of the Study

Assessment Module

The functionality of providing exercises designed for the learning of topics and concepts in programming and evaluation of the results are handled by this module

Related Studies

- Prototype of Visual Programming Environment for C Language Novice Programmer (Abe, K., Fukawa, Y., & Tanaka, T., 2019)
- On the Design of a Generic Visual Programming Environment (Zhang, D.-Q., & Zhang, K.)
- Environment pi J for Visual Programming in Java (Prokhorov, V., & Kosarev, V., 1999)
- HASKEU: An editor to support visual and textual programming in tandem (Alam, A., & Bush, V. , 2016)
- The Scratch Programming Language and Environment (Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. ,2010)

Methodology - Materials

Sofware Requirements - Development

- Linux 5.4 kernel with Manjaro distribution as Operating System
- Terminal for running commands
- Vim for text and code editing
- Kha for graphical and media framework
- zui for user-interface base framework
- Haxe programming language

Methodology - Materials

Hardware Requirements - Development

- Laptop
- 2GB RAM (Random Access Memory)
- Intel Core 2 Duo at 1.4 GHz processor
- 80 GB HDD

Methodology - Materials

Sofware Requirements - Deployment

- Microsoft Windows 7 or above
- C++ Runtime libraries
- Haxe programming language and enviromnet

Methodology - Materials

Hardware Requirements - Deployment

- Laptop or Desktop
- 2GB RAM (Random Access Memory)
- Atleast Intel Core 2 Duo at 1.4 GHz processor
- Atleast 1 GB HDD

Methodology - Method

V-Model

This model follows the relationships between each of the different phases in the life cycle of the development process, each with an associated testing phase.
The primary focus and purpose of this model is to improve the efficiency of development and to ensure the effectiveness of the software.

Methodology - V-Model

Phases of V-Model

- Requirements
- System Design
- Architecture Design
- Module Design
- Implementation and Coding
- Testings

Methodology - V-Model

Requirements

- Conduction of survey
- Gathering of data
- Conduction of assesment

The end results are:

- Ishikawa Diagrams
- Graphical representations of data

Methodology - V-Model - Requirements

Examples of questions:

```
int a = 20;
int *b = &a;
(*b)++;
1. What is the final of ‘b‘
in line 2?
2. What is the final of ‘b‘
in line 3?
```

```
int a = 5;
int b = 20;
for (int i=a; i<b; i+=2)
{
a++;
b--;
}
1. What is the final value of
‘a‘?
2. What is the final value of
‘b‘?
```

Methodology - V-Model

System Design

- Assessment of the current manual or system
- Schedule of the development

The end results are:

- Context Diagram (Manual)
- Gantt Chart

Methodology - V-Model

Architecture Design

- Specifications as blueprint of the software
- Selection of libraries and tools to be used

The end result is:

- Context Diagram

Methodology - V-Model

Module Design

- Identification and definition of each module
- Scope and integration of the modules to the system

The end result is:

- Theoretical Framework

Methodology - V-Model

Implementation and Coding

- Start of programming and development
- Compilation and running of the modules
- Integration and coupling of the modules as a software

The end results are:

- Modules

Methodology - V-Model

Testing

- Application of tests
- Fixing of bugs, errors, and misbehaviors

To make sure of the following for the quality of the software:

- Functionality
- Efficiency
- Usability
- Portability
- Reliability

Methodology - V-Model

Evaluation

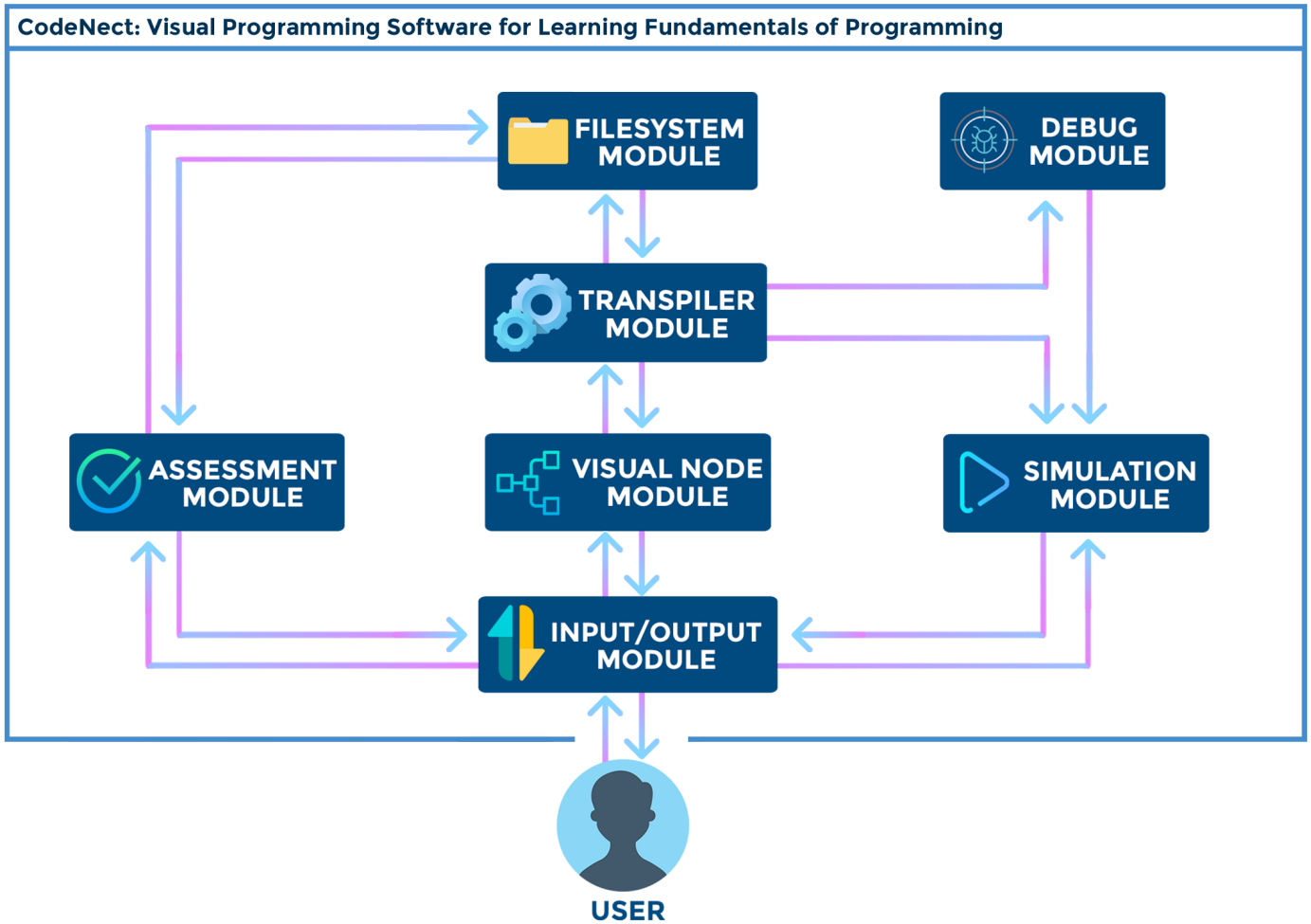
The ISO/IEC/IEEE 29119-4:2015 - Software and systems engineering — Software testing is a a set of five standards for software testing internationally recognized and approved. It was first developed in year 2007 and was released in year 2013. This standard defines the following for usage with software development lifecycle: vocabulary, processes, documentation, techniques, and a process assessment model for testing.

Methodology - V-Model - Evaluation

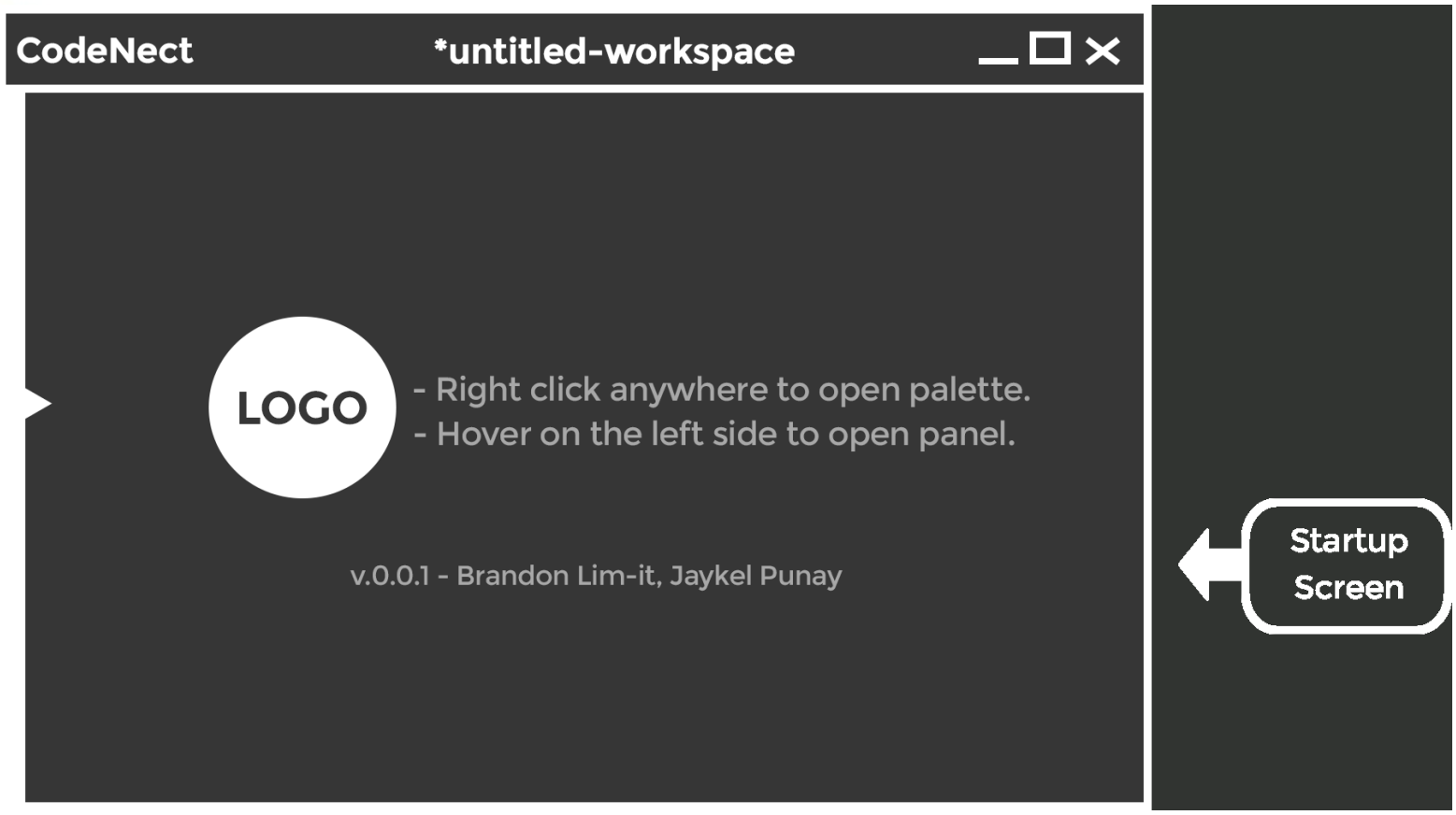
Evaluation

- Respondents will be tasked to solve simple coding exercises
- Respondents will be given a feedback form for assessment
- Assessment of the respondents' experience with using the software
- Evaluation of their solution/answer to coding exercises
- Comparision of the results

System Architecture

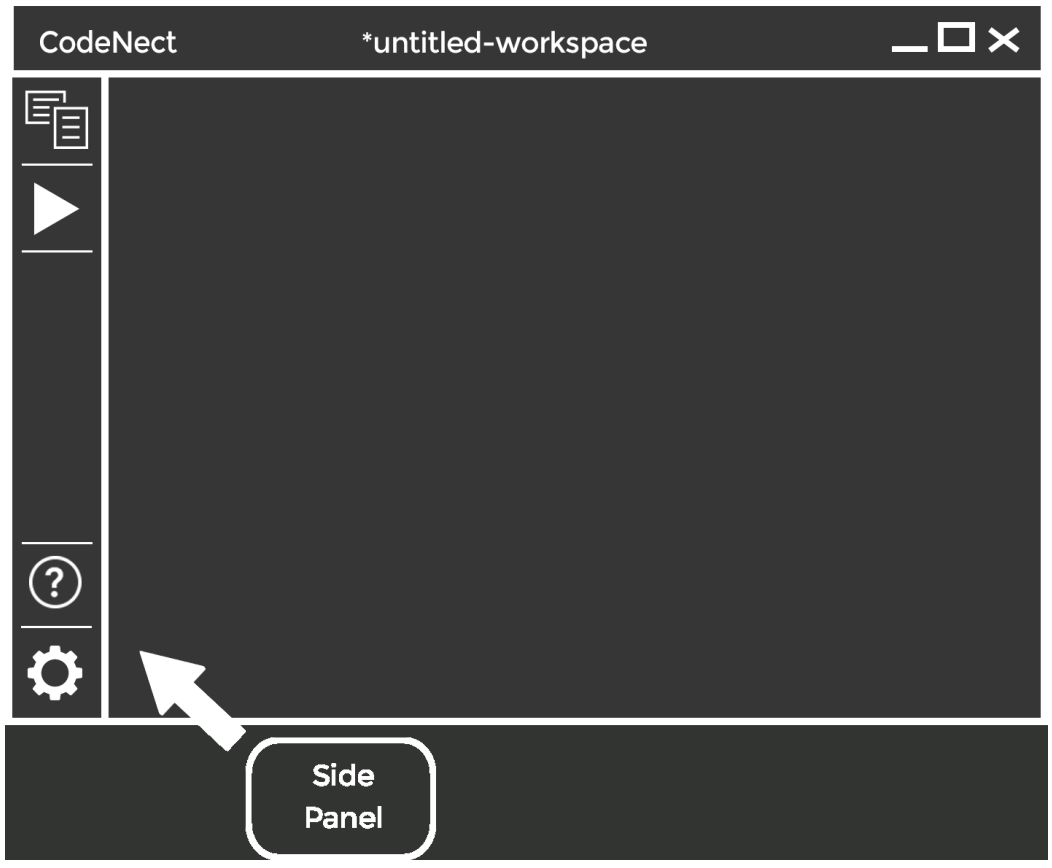


User-Interface Mockup and Design

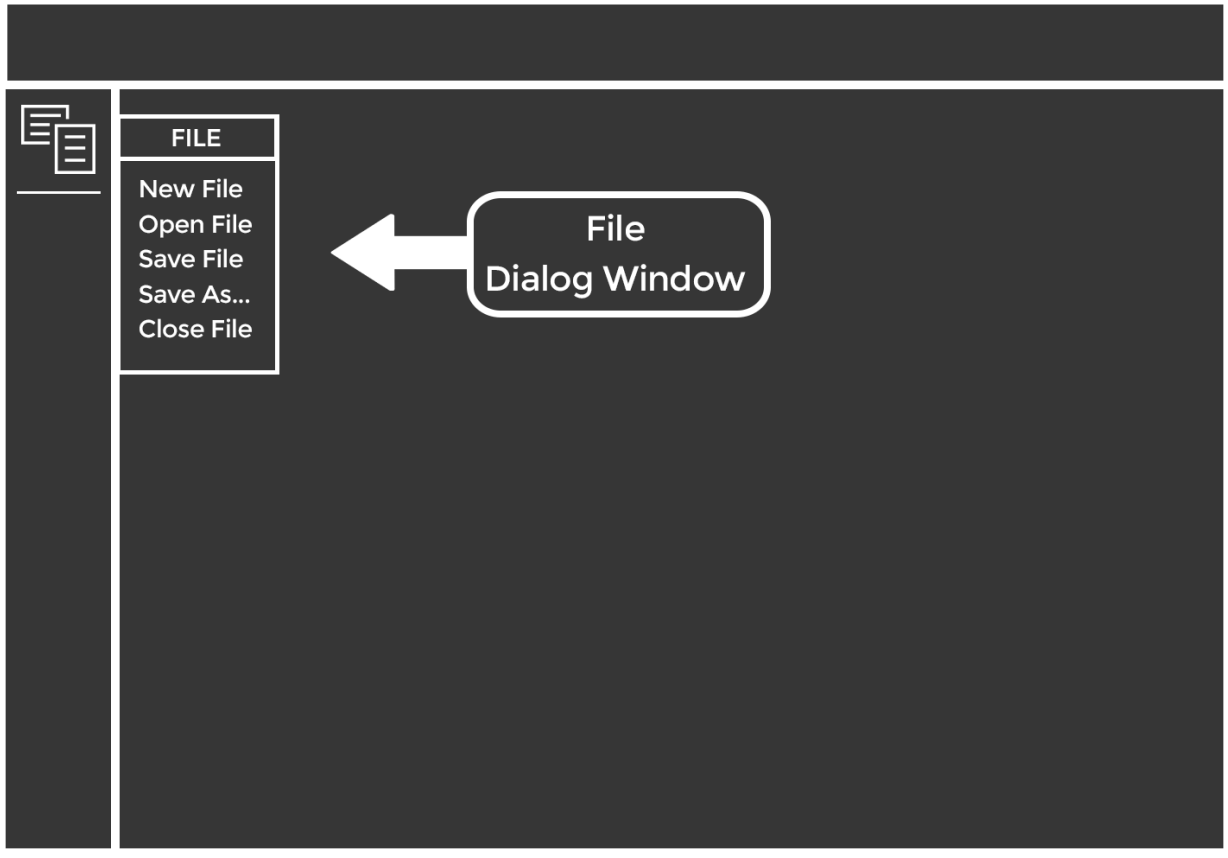


User-Interface Mockup and Design

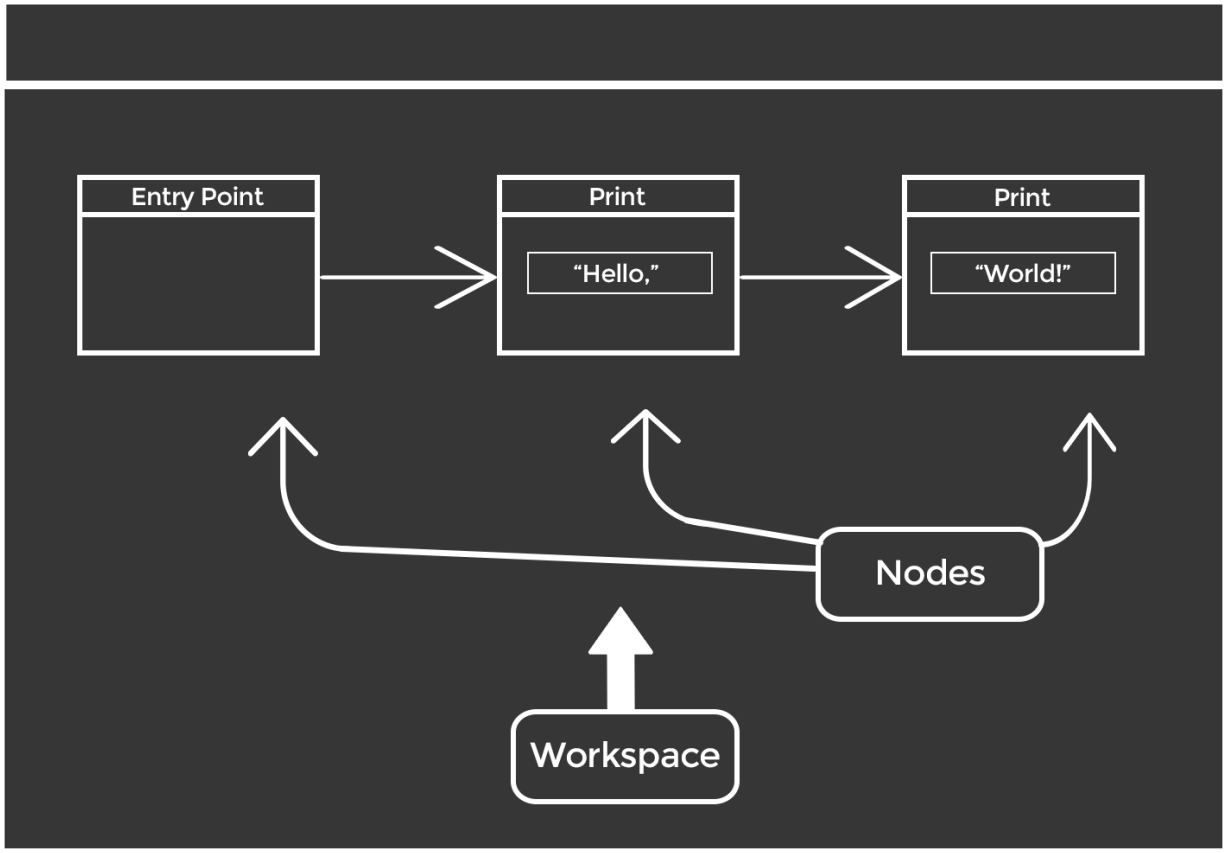
User-Interface Mockup and Design



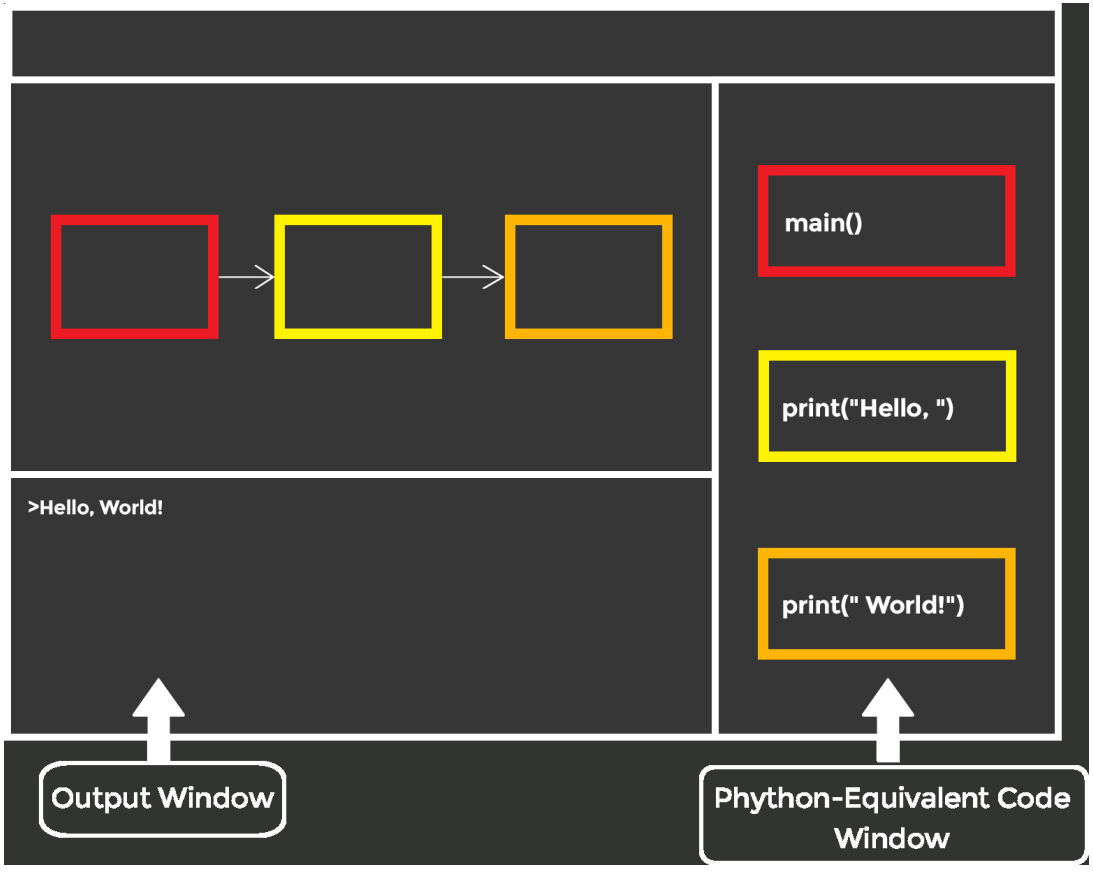
User-Interface Mockup and Design



User-Interface Mockup and Design



User-Interface Mockup and Design



Thank you so much and God bless all!