19AIE112: Elements of Computing Part 2

Lab sheet 3: High-Level Programming (Jack)

Objective of this Lab:

To present an idea of a Jack based Application as a part of the **EndSem Project** of the course. Also, please illustrate how you will implement the Jack Application using pseudo – code by giving a brief outline of the classes, subroutines and other OS elements which will be used.

Following are the guidelines to be followed for the Project -

Instructions

- The Jack Application can be simple but non-trivial Object-Oriented based design involving various concepts of Application design.
- The Jack Application must have minimum of 2 classes
- The Jack OS includes 8 classes: Screen, Memory, String, Array, Sys, Keyboard, Math, and Output.
  Your Jack Application must use APIs of minimum **5** of the above OS classes
- The Jack Application must include all elements of the Jack Syntax Specification – keywords, symbols, constants, identifiers

Deadline of Submission of Project Idea as part of lab sheet

(On or before June 11 before class starts)

Resources : You can refer this and use this for Project Jack Application development

- The relevant reading for this project is book chapter 9. You will need three tools: a text editor for writing your Jack programs, the supplied Jack Compiler for compiling them, and the supplied VM emulator for executing and testing them.

19AIE112: Elements of Computing Part 2

Idea

To implement a simple stack game using the jack programming language

Introduction

Stack is a simple game where the goal is to build a stack of blocks as high as possible.

At each level, a row of blocks moves sideways and the user has to place the blocks in place and timing it so that it aligns with the previous level. Blocks that don't align are lost and if no blocks aligned at all, the player loses the game. As the stack levels increases, the blocks move faster and faster making the timing even more challenging.

Pseudo Codes

```
class Main
{
    function main()
    {
        var Stack stack
        game.run() - execute the game
        game.dispose() - Deallocating the memory
        return
    }
}
```

19AIE112: Elements of Computing Part 2

```
class Constants
{
    function coloumns()
    {
        returns the number of coloumns
    }
    function levels()
    {
        returns the number of levels
    }
    function Space
    {
        return 32 - ascii code of the space key
    }
    function Playgame_P
    {
        return 80 - ascii code of the letter P
    }
    function Quitgame_Q
    {
        return 81 - ascii code of the letter Q
    }
}
```

```
class Draw
{
    function grid()
    {
        Implementing the grid lines
        Screen.setcolor()
        Screen.drawline()
        return
    }

    function row()
    {
        Implementing a row with block arrangement
    }

    function block ( position parameters )
    {
        Draws the block in the arbitrary position
        Screen.drawRectangle()
        Setting colour for the block
        Screen.setColor()
    }
}
```

```
class RowChange
{
    Implement the row moving each sideways

    method setRow ( parameters )
    {
        setting the number of blocks from index
    }

    method getRow()
    {
        return row
    }

    method setLevel()
    {
        Increases speed of blocks for consecutive levels
    }

    method setDelay()
    {
        Initializing the speed of the blocks in each levels
    }

    method move()
    {
        Determines to move the block each sides
    }
}
```

```
class stack
{
    Implementing the stack game

    constructor new()
    {
        return the required coloumn and level values
    }

    method add()
    {
        Adds new row to the stack
    }

    method getRow()
    {
        return the stack level
    }

    method dispose()
    {
        Deallocating the memory components
        Memory.deAlloc()
        return
    }
}
```

```
class gamecontroller
{
    constructor gamecontroller new()
    {
        RowChange.new()
        stack.new()
        return
    }

    method run()
    {
        Running from initial part of the game
        Manipulating the input key
    }

    method gamestate ()
    {
        Finding the state of the game
        Manipulating the level & block count
        return
    }
}
```

19AIE112: Elements of Computing Part 2

```
class game
{
    constructor game()
    {
        Implementing the menu of the game
        return
    }

    method run()
    {
        Running the game
        game.run()
        game.dispose()
        return;
    }
}
```

Group 4 Members

Name	Roll No.
Abhijith A Thampi	AM.EN.U4AIE20102
Ajay G Nair	AM.EN.U4AIE20108
Anagha R	AM.EN.U4AIE20112
Devika S Menon	AM.EN.U4AIE20120
Megha R	AM.EN.U4AIE20147
Neha B	AM.EN.U4AIE20152