Name:_____

Date: _____

The Jack Programming Language

Standard Class Library

```
class Output {
   function void moveCursor(int i, int j)
   function void printChar(char c)
   function void printString(String s)
   function void printInt(int i)
   function void println()
   function void backSpace()
}
```

```
Class Keyboard {
   function char keyPressed()
   function char readChar()
   function String readLine(String message)
   function int readInt(String message)
}
```

```
Class Screen {
   function void clearScreen()
   function void setColor(boolean b)
   function void drawPixel(int x, int y)
   function void drawLine(int x1, int y1, int x2, int y2)
   function void drawRectangle(int x1, int y1, int x2, int y2)
   function void drawCircle(int x, int y, int r)
}
```

```
Class Array {
   function Array new(int size)
   method void dispose()
}
```

```
Class Sys {
   function void halt():
   function void error(int errorCode)
   function void wait(int duration)
}
```

```
Class String {
   constructor String new(int maxLength)
   method void   dispose()
   method int    length()
   method char   charAt(int j)
   method void   setCharAt(int j, char c)
   method String appendChar(char c)
   method void   eraseLastChar()
   method int    intValue()
   method void   setInt(int j)
   function char backSpace()
   function char doubleQuote()
   function char newLine()
}
```

```
class Math {
   function void init()
   function int abs(int x)
   function int multiply(int x, int y)
   function int divide(int x, int y)
   function int min(int x, int y)
   function int max(int x, int y)
   function int sqrt(int x)
}
```

Syntax: keywords

```
/** Procedural processing example */
class Main {
   /* Inputs some numbers and computes their average */
   function void main() {
     var Array a;
     var int length;
     var int i, sum;
     let length = Keyboard.readInt("How many numbers? ");
     let a = Array.new(length); // constructs the array
     let i = 0;
     while (i < length) {
        let a[i] = Keyboard.readInt("Enter a number: ");
        let sum = sum + a[i];
        let i = i + 1;
     }
     ...
}
```

Syntax elements:

- White space / comments
- keywords
- Symbols
- Constants
- Identifiers

class, constructor, method, function	Program components
int, boolean, char, void	Primitive types
var, static, field	Variable declarations
let, do, if, else, while, return	Statements
true, false, null	Constant values
this	Object reference

Syntax: symbols

```
/** Procedural processing example */
class Main {
   /* Inputs some numbers and computes their average */
   function void main() {
     var Array a;
     var int length;
     var int i, sum;
     let length = Keyboard.readInt("How many numbers? ");
     let a = Array.new(length); // constructs the array
     let i = 0;
     while (i < length) {
        let a[i] = Keyboard.readInt("Enter a number: ");
        let sum = sum + a[i];
        let i = i + 1;
     }
     ...
}
```

Syntax elements:

- White space / comments
- keywords
- Symbols
- Constants
- Identifiers

() Used for grouping arithmetic expressions and for enclosing parameter-lists and argument-lists;	, Variable list separator;
[] Used for array indexing;	; Statement terminator;
{} Used for grouping program units and statements;	= Assignment and comparison operator;
	. Class membership;
	+ - * / & | ~ < > Operators.

1. Translate the following Java code into its equivalent Jack code.

```
int x = 3;
int y = 5;
int greatest;
if (x > y)
    greatest = x;
else
    greatest = y;
System.out.println(greatest);
```

Jack Code

2.

```
// Multiplies x * y
// (by summing x, y times)
int x = 2;
int y = 5;
int product = 0;
int n = 1;
while (n <= y)
{
    product += x;
    n++;
}
System.out.println("The product is "
        + product);
```

Jack Code

3. Write the Jack code that produces the following transaction with the user.
   Note: the green text indicates input from the user.

```
Please enter number 1: 21
Please enter number 2: 19
Please enter number 3: 50

The average of the three numbers is: 30
```

Jack Code

4. Write the Jack code that produces the following transaction with the user.
   Note: the green text indicates input from the user.

```
Please enter your birth year...
1934
You are 86 years old.

Please enter a future age...
100
You will be 100 in the year 2120.
```

Jack Code

5. Translate the entire Main class (written in Java) into its Jack equivalent.

```java
public class Main {

    public static void main(String[] args)
    {
        System.out.println(mult(5, 4));
    }

    static int mult(int x, int y)
    {
        int sum = 0;
        int n = 1;
        while (n <= y)
        {
            sum += x;
            n++;
        }
        return sum;
    }
}
```

Jack Code

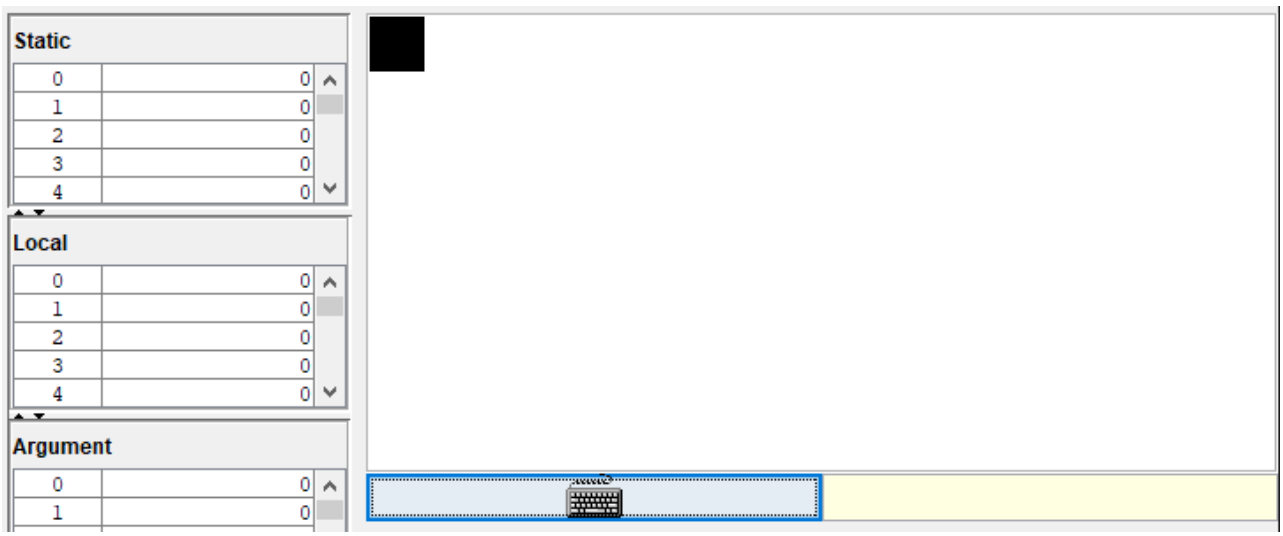
6. Convert the following Java class (Fraction.java) into its Jack equivalent (two files/classes named Fraction.jack and Main.jack).

```java
public class Fraction {

    private int numerator, denominator;
    Fraction(int x, int y) {
        numerator = x;
        denominator = y;
    }
    int getNumerator() { return numerator; }
    int getDenominator() { return denominator; }

    void print() {
        System.out.println(numerator + "/" + denominator);
    }

    Fraction plus(Fraction other) {
        // Cross-multiply, no reduction/simplification
        int num = numerator * other.denominator +
                  other.numerator * denominator;
        int den = denominator * other.denominator;
        return new Fraction(num, den);
    }

    public static void main(String[] args) {
        Fraction f1, f2, f3;
        f1 = new Fraction(2, 3);
        f2 = new Fraction(1, 5);
        f3 = f1.plus(f2);
        f3.print();
    }
}
```

Jack Code

7. Write a Jack program that will display a square (30 pixels x 30 pixels), starting in the top-left (x=0, y=0) of the screen, then moving around the edge of the screen clockwise (e.g. along the top edge, then right edge, bottom edge, left edge) until it gets back to the origin.

   The square should move by itself, with a short wait between each movement.



Jack Code

Summative Questions:

8. When we execute a Jack program, the first subroutine that starts running is:

9. Can a subroutine in one Jack class access field variables of another Jack class?

10. Which Jack classes should have a method for disposing objects?

11. What does the keyword “this” implicitly refer to? (Select all that apply)
    a) In constructors: the current object
    b) In functions: the current object
    c) In methods: the current object
    d) In Main.main: the current object

12. Which of the following are true about Jack classes? (Select all that apply)
    a) A Jack class must have a constructor
    b) A Jack class can contain either methods, or functions, but not both
    c) Each Jack class must be stored in a separate file
    d) Each Jack class must have a subroutine named “main”