Software Engineering (IT-314)

ID: 202201238

Name: Riddhi Mistry

Lab – 7

---

1. Armstrong

A. Program Inspection

1. 2 errors. Remainder = num%10 and num = num/10

```
while(num > 0){
    remainder = num / 10;
    check = check + (int)Math.pow(remainder,3);
    num = num % 10;
}
```

2. Most effective category is C: Computation Errors
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 2 errors. Remainder = num%10 and num = num/10
2. For fixing the error I need 2 break point at line number 10 and 12. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.
3.

```java
//Armstrong Number
class Armstrong{
    public static void main(String args[]){
        int num = Integer.parseInt(args[0]);
        int n = num; //use to check at last time
        int check=0,remainder;
        while(num > 0){
            remainder = num % 10;
            check = check + (int)Math.pow(remainder,3);
            num = num / 10;
        }
        if(check == n)
            System.out.println(n+" is an Armstrong Number");
        else
            System.out.println(n+" is not a Armstrong Number");
    }
}
```

2. GCD and LCM

A. Program Inspection

1. 3 errors. a = (x>y) ? x:y; , while(a%b!=0) and in LCM while loop if condition change if(a%x==0&&a%y==0)

```java
        int r=0, a, b;
        a = (x > y) ? y : x; // a is greater number 1
        b = (x < y) ? x : y; // b is smaller number

    while(a % b == 0) //Error replace it with while(a % b != 0) 2
    {
        r = a % b;
        a = b;
        b = r;
    }
    return r;
}
```

```
27        while(true)
28        {
29            if(a % x != 0 && a % y != 0) // 3
30                return a;
31            ++a;
32        }
```

2. Most effective category is C: Computation Errors.
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 3 errors. a = (x>y) ? x:y; , while(a%b!=0) and in LCM while loop if condition change if(a%x==0&&a%y==0)
2. For fixing the error I need 3 break point at line number 11,14 and 28. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.
3.

```
6   public class GCD_LCM
7   {
8       static int gcd(int x, int y)  1 usage
9       {
10          int r=0, a, b;
11          a = (x > y) ? x : y; // a is greater number 1
12          b = (x < y) ? x : y; // b is smaller number
13          r = b;
14          while(a % b != 0) //Error replace it with while(a % b != 0) 2
15          {
16              r = a % b;
17              a = b;
18              b = r;
19          }
20          return r;
21      }
22
23      static int lcm(int x, int y)  1 usage
24      {
25          int a;
26          a = (x > y) ? x : y; // a is greater number
27          while(true){
28              if(a % x == 0 && a % y == 0) // 3
29                  return a;
30              ++a;
31          }
32      }
```

3. Knapsack

A. Program Inspection

1. 2errors. Option1 = opt[n-1][w],weight[n]<=w

```
25
26                  // don't take item n
27                  int option1 = opt[n++][w];
```

```
29          // take item n
30          int option2 = Integer.MIN_VALUE;
31          if (weight[n] > w) option2 = profit[n-2] + opt[n-1][w-weight[n]];
```

2. Most effective category is C: Computation Errors.
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 2errors. Option1 = opt[n-1][w],weight[n]<=w
2. For fixing the error I need 2 break point at line number 28 and 32. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.
3.

```java

    for (int n = 1; n <= N; n++) {
        for (int w = 1; w <= W; w++) {

            // don't take item n
            int option1 = opt[n-1][w];

            // take item n
            int option2 = Integer.MIN_VALUE;
            if (weight[n] <= w) option2 = profit[n] + opt[n-1][w-weight[n]]

            // select better of two options
            opt[n][w] = Math.max(option1, option2);
            sol[n][w] = (option2 > option1);
        }
    }

```

4. Magic Number Check

A. Program Inspection

1. 5errors, s=1,sum>0,s=s*(sum%10),sum=sum/10;,semicon at line 19.

```java
15      sum=num;int s=0;
16      while(sum==0)
17      {
18          s=s*(sum/10);
19          sum=sum%10
20      }
```

2. Most effective category is C: Computation Errors and category E: Control-Flow Errors and B: Data-Declaration Errors.
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 5errors, s=1,sum>0,s=s*(sum%10),sum=sum/10;,semicon at line 19.
2. For fixing the error I need 3 break point at line number 16,18 and 19. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.

```
5  public class MagicNumberCheck
6  {
7      public static void main(String args[])
8      {
9          Scanner ob=new Scanner(System.in);
10         System.out.println("Enter the number to be checked.");
11         int n=ob.nextInt();
12         int sum=0,num=n;
13         while(num>9)
14         {
15             sum=num;int s=1;
16             while(sum>0)
17             {
18                 s=s*(sum%10);
19                 sum=sum/10;
20             }
21             num=s;
22         }
23         if(num==1)
24         {
25             System.out.println(n+" is a Magic Number.");
26         }
27         else
28         {
29             System.out.println(n+" is not a Magic Number.");
30         }
31     }
32 }
```

3.

5.Merge Sort

A. Program Inspection

1. 3errors, int[] left = leftHalf(array), int[] right = rightHalf(array),merge(array,left,right)

```
22        int[] left = leftHalf( array: array+1);
23        int[] right = rightHalf( array: array-1);
```

```
29        // merge the sorted halves into a sorted whole
30        merge(array, left++, right--);
```

2. Most effective category F: Interface Errors
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 3errors, int[] left = leftHalf(array), int[] right = rightHalf(array),merge(array,left,right)
2. For fixing the error I need 3 break point at line number 23,24 and 31. For fixing observing the output and after adding checkpoints at the code what should we

expect and what code give this identify by observation.

```java

public class MergeSort {
    public static void main(String[] args) {
        int[] list = {14, 32, 67, 76, 23, 41, 58, 85};
        System.out.println("before: " + Arrays.toString(list));
        mergeSort(list);
        System.out.println("after:  " + Arrays.toString(list));
    }

    // Places the elements of the given array into sorted order
    // using the merge sort algorithm.
    // post: array is in sorted (nondecreasing) order
    public static void mergeSort(int[] array) {  3 usages
        if (array.length > 1) {
            // split array into two halves

            int[] left = leftHalf(array);
            int[] right = rightHalf(array);

            // recursively sort the two halves
            mergeSort(left);
            mergeSort(right);

            // merge the sorted halves into a sorted whole
            merge(array, left, right);
        }
    }

    // Returns the first half of the given array.
    public static int[] leftHalf(int[] array) {  1 usage
        int size1 = array.length / 2;
        int[] left = new int[size1];
        for (int i = 0; i < size1; i++) {
            left[i] = array[i];
        }
```

```java
        return left;
    }

    // Returns the second half of the given array.
    public static int[] rightHalf(int[] array) {  1 usage
        int size1 = array.length / 2;
        int size2 = array.length - size1;
        int[] right = new int[size2];
        for (int i = 0; i < size2; i++) {
            right[i] = array[i + size1];
        }
        return right;
    }

    // Merges the given left and right arrays into the given
    // result array.  Second, working version.
    // pre : result is empty; left/right are sorted
    // post: result contains result of merging sorted lists;
    public static void merge(int[] result,  1 usage
                             int[] left, int[] right) {
        int i1 = 0;   // index into left array
        int i2 = 0;   // index into right array

        for (int i = 0; i < result.length; i++) {
            if (i2 >= right.length || (i1 < left.length &&
                    left[i1] <= right[i2])) {
                result[i] = left[i1];    // take from left
                i1++;
            } else {
                result[i] = right[i2];   // take from right
                i2++;
            }
        }
    }
}
```

6.Matrix Multiplication

A. Program Inspection

1. 2errors, k<n and sum = sum + first[c][k]*second[k][d]

```
45      {
46          for ( k = 0 ; k < p ; k++ )
47          {
            sum = sum + first[c-1][c-k]*second[k-1][k-d];
49          }
```

2. Most effective category is C: Computation Errors and category E: Control-Flow Errors.
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 2errors, k<n and sum = sum + first[c][k]*second[k][d]
2. For fixing the error I need 2 break point at line number 46 and 48. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.

3.

```java
//Java program to multiply two matrices
import java.util.Scanner;

class MatrixMultiplication
{
    public static void main(String args[])
    {
        int m, n, p, q, sum = 0, c, d, k;

        Scanner in = new Scanner(System.in);
        System.out.println("Enter the number of rows and columns of first matrix");
        m = in.nextInt();
        n = in.nextInt();

        int first[][] = new int[m][n];

        System.out.println("Enter the elements of first matrix");

        for ( c = 0 ; c < m ; c++ )
            for ( d = 0 ; d < n ; d++ )
                first[c][d] = in.nextInt();

        System.out.println("Enter the number of rows and columns of second matrix");
        p = in.nextInt();
        q = in.nextInt();

        if ( n != p )
            System.out.println("Matrices with entered orders can't be multiplied with each other.");
        else
        {
            int second[][] = new int[p][q];
            int multiply[][] = new int[m][q];

            System.out.println("Enter the elements of second matrix");

            for ( c = 0 ; c < p ; c++ )
                for ( d = 0 ; d < q ; d++ )
                    second[c][d] = in.nextInt();

            for ( c = 0 ; c < m ; c++ )
```

```java
            {
                for ( d = 0 ; d < q ; d++ )
                {
                    for ( k = 0 ; k < n ; k++ )
                    {
                        sum = sum + first[c][k]*second[k][d];
                    }

                    multiply[c][d] = sum;
                    sum = 0;
                }
            }

            System.out.println("Product of entered matrices:-");

            for ( c = 0 ; c < m ; c++ )
            {
                for ( d = 0 ; d < q ; d++ )
                    System.out.print(multiply[c][d]+"\t");

                System.out.print("\n");
            }
        }
    }
}
```

7.Quadratic Probing Hash-Table Test

A. Program Inspection

1. 1error, Insert function: i = (i + h * h++) % maxSize;

```
69          }
            i += (i + h / h--) % maxSize;
```

5. Most effective category is C: Computation Errors
6. Input / Output Errors, because if user enter “qw23” then this is wrong input
7. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 1error, Insert function: i = (i + h * h++) % maxSize;
2. For fixing the error I need 1 break point at line number 70. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.

```java
class QuadraticProbingHashTable  2 usages

{

    private int currentSize, maxSize;  7 usages
    private String[] keys;  14 usages
    private String[] vals;  9 usages
    public QuadraticProbingHashTable(int capacity)  1 usage
    {
        currentSize = 0;
        maxSize = capacity;
        keys = new String[maxSize];
        vals = new String[maxSize];
    }
    /** Function to clear hash table **/
    public void makeEmpty()  1 usage
    {
        currentSize = 0;
        keys = new String[maxSize];
        vals = new String[maxSize];
    }
    /** Function to get size of hash table **/
    public int getSize()  2 usages
    {
        return currentSize;
    }
    /** Function to check if hash table is full **/
    public boolean isFull()  no usages
    {
        return currentSize == maxSize;
    }
    /** Function to check if hash table is empty **/
    public boolean isEmpty()  no usages
    {
        return getSize() == 0;
    }
```

3.

```java
    }
    /** Fucntion to check if hash table contains a key **/
    public boolean contains(String key)  1 usage
    {
        return get(key) !=  null;
    }
    /** Functiont to get hash code of a given key **/
    private int hash(String key)  3 usages
    {
        return key.hashCode() % maxSize;
    }
    /** Function to insert key-value pair **/
    public void insert(String key, String val)  2 usages
    {
        int tmp = hash(key);
        int i = tmp, h = 1;
        do
        {
            if (keys[i] == null)
            {
                keys[i] = key;
                vals[i] = val;
                currentSize++;
                return;
            }
            if (keys[i].equals(key))
            {
                vals[i] = val;
                return;
            }
            i = (i + h * h++) % maxSize;

        } while (i != tmp);
    }
    /** Function to get value for a given key **/
    public String get(String key)  2 usages
    {
        int i = hash(key), h = 1;
        while (keys[i] != null)
```

```java
                return vals[i];
                i = (i + h * h++) % maxSize;
            System.out.println("i "+ i);
        }
        return null;
    }
    /** Function to remove key and its value **/
    public void remove(String key)  1 usage
    {
        if (!contains(key))
            return;
        /** find position key and delete **/
        int i = hash(key), h = 1;
        while (!key.equals(keys[i]))
            i = (i + h * h++) % maxSize;
        keys[i] = vals[i] = null;
        /** rehash all keys **/
        for (i = (i + h * h++) % maxSize; keys[i] != null; i = (i + h * h++) % maxSize)
        {
            String tmp1 = keys[i], tmp2 = vals[i];
            keys[i] = vals[i] = null;
            currentSize--;
            insert(tmp1, tmp2);
        }
        currentSize--;
    }
    /** Function to print HashTable **/
    public void printHashTable()  1 usage
    {
        System.out.println("\nHash Table: ");
        for (int i = 0; i < maxSize; i++)
            if (keys[i] != null)
                System.out.println(keys[i] +" "+ vals[i]);
        System.out.println();
    }
}
```

```java
public class QuadraticProbingHashTableTest
{
    public static void main(String[] args)
    {
        Scanner scan = new Scanner(System.in);
        System.out.println("Hash Table Test\n\n");
        System.out.println("Enter size");
        /** maxSizeake object of QuadraticProbingHashTable **/
        QuadraticProbingHashTable qpht = new QuadraticProbingHashTable(scan.nextInt() );
        char ch;
        /**  Perform QuadraticProbingHashTable operations  **/
        do
        {
            System.out.println("\nHash Table Operations\n");
            System.out.println("1. insert ");
            System.out.println("2. remove");
            System.out.println("3. get");
            System.out.println("4. clear");
            System.out.println("5. size");
            int choice = scan.nextInt();
            switch (choice)
            {
                case 1 :
                    System.out.println("Enter key and value");
                    qpht.insert(scan.next(), scan.next() );
                    break;
                case 2 :
                    System.out.println("Enter key");
                    qpht.remove( scan.next() );
                    break;
                case 3 :
                    System.out.println("Enter key");
                    System.out.println("Value = "+ qpht.get( scan.next() ));
                    break;
                case 4 :
                    qpht.makeEmpty();
                    System.out.println("Hash Table Cleared\n");
                    break;
                case 5 :
                    System.out.println("Size = "+ qpht.getSize() );
                    break;
```

```java
            default :
                System.out.println("Wrong Entry \n ");
                break;
        }
        /** Display hash table **/
        qpht.printHashTable();

        System.out.println("\nDo you want to continue (Type y or n) \n");
        ch = scan.next().charAt(0);
    } while (ch == 'Y'|| ch == 'y');
  }
}
```

8.Ascending Order

A. Program Inspection

1. 4error, use class name AscendingOrder, for(int i=0;i<n;i++) and remove semicolon, change if condition if(a[i]>a[j])

```java
public class Ascending _Order
{
```

```java
    }
    for (int i = 0; i >= n; i++);
    {
```

```java
        if (a[i] <= a[j])
```

2. Most effective category is D: Comparison Errors, E: Control-Flow Errors

3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 4error, use class name AscendingOrder, for(int i=0;i<n;i++) and remove semicolon, change if condition if(a[i]>a[j])
2. For fixing the error I need 2 break point at line number 15 and 23. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.

```
4   import java.util.Scanner;
5   public class AscendingOrder
6   {
7       public static void main(String[] args)
8       {
9           int n, temp;
10          Scanner s = new Scanner(System.in);
11          System.out.print("Enter no. of elements you want in array:");
12          n = s.nextInt();
13          int a[] = new int[n];
14          System.out.println("Enter all the elements:");
15          for (int i = 0; i < n; i++)
16          {
17              a[i] = s.nextInt();
18          }
19          for (int i = 0; i < n; i++)
20          {
21              for (int j = i + 1; j < n; j++)
22              {
23                  if (a[i] > a[j])
24                  {
25                      temp = a[i];
26                      a[i] = a[j];
27                      a[j] = temp;
28                  }
29              }
30          }
31          System.out.print("Ascending Order:");
32          for (int i = 0; i < n - 1; i++)
33          {
34              System.out.print(a[i] + ",");
35          }
36          System.out.print(a[n - 1]);
37      }
38  }
```

3.

9. Stack Revise Demo

A. Program Inspection

1. 3error, in display function change for loop condition for(int i=0;i<top;i++), change pop function top--; and change push function top++;

```java
for(int i=0;i>top;i++){
    System.out.print(stack[i]+ " ");
}
```

```java
public void pop(){  4 usages
    if(!isEmpty())
        top++;
```

```java
public void push(int value){  5 usages
    if(top==size-1){
        System.out.println("Stack is full, can't push a value");
    }
    else{
        top--;
        stack[top]=value;
    }
}
```

2. Most effective category is Category C: Computation Errors, E: Control-Flow Errors
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 3error, in display function change for loop condition for(int i=0;i<top;i++), change pop function top--; and change push function top++;
2. For fixing the error I need 3 break point at line number 21 , 28 and 40. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.

```
5   class StackMethods {  2 usages
6       private int top;  7 usages
7       int size;  3 usages
8       int[] stack ;  3 usages
9
10      public StackMethods(int arraySize){  1 usage
11          size=arraySize;
12          stack= new int[size];
13          top=-1;
14      }
15
16      public void push(int value){  5 usages
17          if(top==size-1){
18              System.out.println("Stack is full, can't push a value");
19          }
20          else{
21              top++;
22              stack[top]=value;
23          }
24      }
25
26      public void pop(){  4 usages
27          if(!isEmpty())
28              top--;
29          else{
30              System.out.println("Can't pop...stack is empty");
31          }
32      }
33
34      public boolean isEmpty(){  1 usage
35          return top==-1;
36      }
37
38      public void display(){  2 usages
39
40          for(int i=0;i<top;i++){
41              System.out.print(stack[i]+ " ");
42          }
```

3.

```java
        System.out.println();
    }
}

public class StackReviseDemo {

    public static void main(String[] args) {
        StackMethods newStack = new StackMethods( arraySize: 5);
        newStack.push( value: 10);
        newStack.push( value: 1);
        newStack.push( value: 50);
        newStack.push( value: 20);
        newStack.push( value: 90);

        newStack.display();
        newStack.pop();
        newStack.pop();
        newStack.pop();
        newStack.pop();
        newStack.display();
    }
}
```

10. Main Class

A. Program Inspection

1. 2error, missing semicolon and wrong parameters doTowers(topN-1,inter,from,to)

```java
doTowers(topN ++, inter--, inter: from+1, to: to+1)
```

2. Most effective category is Category F: Interface Errors
3. Input / Output Errors, because if user enter “qw23” then this is wrong input
4. Yes, Program inspection is definitely worth applying here: It helped identify logical errors that were leading to incorrect computation. It is a simple and effective way to catch issues early in the code before running it.

B. Debugging

1. 2error, missing semicolon and wrong parameters doTowers(topN-1,inter,from,to)
2. For fixing the error I need 1 break point at line number 18. For fixing observing the output and after adding checkpoints at the code what should we expect and what code give this identify by observation.
3.

```
3   //Tower of Hanoi
4   public class MainClass {
5       public static void main(String[] args) {
6           int nDisks = 3;
7           doTowers(nDisks, from: 'A', inter: 'B', to: 'C');
8       }
9       public static void doTowers(int topN, char from,  3 usages
10                                  char inter, char to) {
11          if (topN == 1){
12              System.out.println("Disk 1 from "
13                      + from + " to " + to);
14          }else {
15              doTowers( topN: topN - 1, from, to, inter);
16              System.out.println("Disk "
17                      + topN + " from " + from + " to " + to);
18              doTowers( topN: topN-1, inter, from, to);
19          }
20      }
21  }
```

Acknowledgement: Was unable to find 2000 lines of code, so answered the questions using small fragments of code that summed up to approximately 2000.

First Segment

Githublink

```
// Copyright 2012 The Obvious Corporation

var fs = require("fs");
var CSS = require("css");
var path = require("path");
var async = require("async");

// Main SUS function that acts as a constructor
module.exports = SUS;

function SUS(source, options) {
    if (!(this instanceof SUS)) return new SUS(source, options);
    this.source = source; // The CSS source code
    this.options = options || {}; // Options for processing
}

// Regular expression constants used throughout the module
SUS.DOT_REGEXP = /^./; // Matches any character
SUS.URL_REGEXP = /url"]+)['"]?\)/; // Matches url() functions
SUS.URL_REGEXP_GLOBAL = /url\s*\(['"]?[^\)'"]+['"]?\)/g; // Global match for
url() functions
SUS.PROTOCOCAL_REGEXP = /\/\//; // Matches protocol-relative URLs

// Utility function to extend an object with properties from other objects
function extend(obj) {
    Array.prototype.slice.call(arguments, 1).forEach(function (source) {
        for (var prop in source) {
            obj[prop] = source[prop];
        }
    });
    return obj;
}



// Function to parse CSS rules, generating sprite images where applicable
function parseRules(base, sprites, options, complete) {
    // Filter out empty rules and process each rule
    async.filterSeries(base.rules, function (rule, nextRule) {
        // Handle media queries recursively
        if (rule.rules) {
```

```
            var spriteKey = sprites.rules.length; // Store the index for
potential removal
            var _sprite = extend({}, rule);
            _sprite.rules = [];
            sprites.rules.push(_sprite); // Push new sprite object to sprites

            return parseRules(rule, _sprite, options, function (err, result) {
                if (err) return nextRule(err); // Handle error in parsing
rules
                if (!_sprite.rules.length) sprites.rules.splice(spriteKey, 1);
// Remove empty sprite
                nextRule(rule.rules = result.length && result); // Set the
parsed rules
            });
        }


        // Use setImmediate to prevent exceeding call stack limit on large CSS
files
        setImmediate(function () {
            // Skip keyframes and font-face declarations
            if (rule.keyframes || (rule.selectors[0] && rule.selectors[0] ==
'@font-face')) {
                return nextRule(rule);
            }

            // Filter declarations for those containing a url()
            async.filterSeries(rule.declarations, function (declaration,
nextDeclaration) {
                var files = declaration.value.match(SUS.URL_REGEXP_GLOBAL);
                // Exit early if declaration doesn't contain a URL
                if (!files) return nextDeclaration(declaration);

                // Map over each found URL and process it
                async.map(files, function (file, nextFile) {
                    var filepath = file.match(SUS.URL_REGEXP)[1];

                    // Exit early if URL is remote
                    if (SUS.PROTOCOCAL_REGEXP.test(filepath)) return
nextFile(null);

                    // Resolve the base path for the file
                    if (typeof options.base === 'function') {
                        filepath = options.base(filepath);
                    } else if (typeof options.base !== 'undefined') {
                        filepath = path.join(options.base, filepath);
                    }
```

```
                    // Read the image file and convert it to base64
                    fs.readFile(filepath, "base64", function (err, data) {
                        if (err) {
                            console.error(`Error reading file ${filepath}:
${err.message}`); // Log error
                            return nextFile(null); // Continue processing even
if there's an error
                        }
                        nextFile(null, {
                            expression: file,
                            ext:
path.extname(filepath).replace(SUS.DOT_REGEXP, ""),
                            path: filepath,
                            data: data
                        });
                    });


                }, function (err, results) {
                    if (err) {
                        console.error(`Error processing files:
${err.message}`); // Log error
                        return nextDeclaration(declaration); // Continue even
on error
                    }
                    // Filter out any null results
                    results = results.filter(r => r);
                    if (!results.length) return nextDeclaration(declaration);
                    parseSprite(results, sprites, rule, declaration,
nextDeclaration);
                });


            }, function (result) {
                nextRule(rule.declarations = result.length && result); //
Update rule declarations
            });
        });
    }, function (result) {
        complete(null, (base.rules = result)); // Complete parsing and return
results
    });
}



// Function to create a sprite rule from the processed files
function parseSprite(files, sprites, rule, declaration, complete) {
```

```
    var value = declaration.value;
    var spriteRule;
    var spriteDeclaration;
    var dataURI;

    // Replace the original file URLs with data URIs
    files.forEach(function (file) {


        dataURI = "url(" + file.data + ")";
        value = value.replace(file.expression, dataURI);


    });

    // Create the new sprite declaration
    spriteDeclaration = {
        "property": declaration.property,
        "value": value
    };

    // Create the new sprite rule
    spriteRule = {
        "selectors": rule.selectors,
        "declarations": [spriteDeclaration]
    };

    // Add the new sprite rule to the sprites collection
    sprites.rules.push(spriteRule);
    complete(declaration); // Complete the processing for this declaration
}

// Main parsing function
SUS.prototype.parse = function (callback) {


    this._base = CSS.parse(this.source); // Parse the CSS source
    this._sprites = { "stylesheet": { "rules": [] } }; // Initialize sprites

    // Start parsing rules and call the callback on completion
    parseRules(this._base.stylesheet, this._sprites.stylesheet, this.options,
function (err) {
        callback(err, this); // Return any error and the current instance
    }.bind(this));

    return this;


};
```

```
// Function to return the base CSS as a string
SUS.prototype.base = function () {

    return CSS.stringify(this._base);

};

// Function to return the sprites as a string

SUS.prototype.sprites = function () {


    return CSS.stringify(this._sprites);
};
```

A. Program Inspection

1. Use Asyn function or arrow function,wrong regex function of url.
2. Category B: Data-Declaration Errors, Category F: Interface Errors
3. Logical errors and input output errors not find using program inspection
4. YES, program inspection is worth applying as it helps in identifying potential issues early in the development process.

Second Segment

GitHub

```
const gameBoard = document.querySelector("#gameboard");
const playerDetails = document.querySelector("#player");
const infoDisplay = document.querySelector("#info-display");
```

```
const err = document.querySelector("#err");
const width = 8

let playerTurn = 'black';
playerDetails.textContent = 'black'

const startPieces = [
    Rook, Knight, Bishop, Queen, King, Bishop, Knight, Rook,
    Pawn, Pawn, Pawn, Pawn, Pawn, Pawn, Pawn, Pawn,
    '', '', '', '', '', '', '', '',
    '', '', '', '', '', '', '', '',
    '', '', '', '', '', '', '', '',
    '', '', '', '', '', '', '', '',
    Pawn, Pawn, Pawn, Pawn, Pawn, Pawn, Pawn, Pawn,
    Rook, Knight, Bishop, King, Queen, Bishop, Knight, Rook,
]



function createBoard() {
    startPieces.forEach((startPiece, i) => {
        const square = document.createElement("div");
        square.classList.add("square");
        square.innerHTML = startPiece

        square.setAttribute("square-id", i);
        square.firstChild?.setAttribute('draggable', true)


        const row = Math.floor((63 - i) / 8) + 1;

        if (row % 2 === 0) {
            square.classList.add(i % 2 == 0 ? "beige" : "brown");
        } else {
            square.classList.add(i % 2 == 0 ? "brown" : "beige");

        }

        if (i <= 15) {
            square.firstChild.firstChild.classList.add("black");
        }
        if (i >= 48) {
            square.firstChild.firstChild.classList.add("white");
        }


        gameBoard.append(square);
    });
};
```

```
createBoard();


const allSquares = document.querySelectorAll("#gameboard .square");
// console.log(allSquares)

allSquares.forEach(square => {
    square.addEventListener('dragstart', dragstart);
    square.addEventListener('dragover', dragover);
    square.addEventListener('drop', dragdrop);
})

let startPositionId
let draggedElement

function dragstart(e) {
    startPositionId = e.target.parentNode.getAttribute("square-id")
    draggedElement = e.target
}

function dragover(e) {
    e.preventDefault();
}

function dragdrop(e) {
    e.stopPropagation();


    // console.log('player go', playerTurn)
    // console.log('target', e.target)
    // console.log(draggedElement)


    const correctTurn =
draggedElement.firstChild.classList.contains(playerTurn);

    const taken = e.target.classList.contains('piece');

    const valid = checkIfValid(e.target);

    const opponentTurn = playerTurn === 'white' ? 'black' : 'white';

    const takenByOpponent =
e.target.firstChild?.classList.contains(opponentTurn);
    // console.log('opp go', opponentTurn)


    if (correctTurn) {
```

```
        // must check this condition
        if (takenByOpponent && valid) {
            e.target.parentNode.append(draggedElement);
            e.target.remove();
            checkForWin();
            changePlayer();
            return
        }
        if (taken && !takenByOpponent) {
            err.textContent = 'Can not go there'
            setTimeout(() => {
                err.textContent = ''
            }, 2000);
            return
        }
        if (valid) {
            e.target.append(draggedElement);
            checkForWin();
            changePlayer();
            return
        }
    }
}

function checkIfValid(target) {
    const targetId = Number(target.getAttribute('square-id')) ||
Number(target.parentNode.getAttribute('square-id'));
    const startId = Number(startPositionId);
    const piece = draggedElement.id
    console.log(startId, targetId, piece)

    switch (piece) {
        case 'pawn':
            const starterRow = [8, 9, 10, 11, 12, 13, 14, 15];
            if (starterRow.includes(startId) && startId + width * 2 ===
targetId ||
                startId + width === targetId ||
                startId + width - 1 === targetId &&
document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild ||
                startId + width + 1 === targetId &&
document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild) {
                return true;
            }
            break;
        case 'knight':
            if (
                startId + width * 2 + 1 === targetId ||
                startId + width * 2 - 1 === targetId ||
```

```
                startId + width - 2 === targetId ||
                startId + width + 2 === targetId ||
                startId - width * 2 + 1 === targetId ||
                startId - width * 2 - 1 === targetId ||
                startId - width + 2 === targetId ||
                startId - width - 2 === targetId
            ) {
                return true
            }
            break;

        case 'bishop':
            if (
                // for right cross --- forward
                startId + width + 1 === targetId ||
                startId + width * 2 + 2 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
||
                startId + width * 3 + 3 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild ||
                startId + width * 4 + 4 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild ||
                startId + width * 5 + 5 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 + 4}"]`).firstChild ||
                startId + width * 6 + 6 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 + 5}"]`).firstChild ||
                startId + width * 7 + 7 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 + 5}"]`).firstChild &&
```

```
!document.querySelector(`[square-id = "${startId + width * 6 +
6}"]`).firstChild ||

                // for left cross --- forward
                startId + width - 1 === targetId ||
                startId + width * 2 - 2 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
||
                startId + width * 3 - 3 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild ||
                startId + width * 4 - 4 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild ||
                startId + width * 5 - 5 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 - 4}"]`).firstChild ||
                startId + width * 6 - 6 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 - 5}"]`).firstChild ||
                startId + width * 7 - 7 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 - 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 -
6}"]`).firstChild ||

                // for right cross --- backward
                startId - width - 1 === targetId ||
                startId - width * 2 - 2 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
||
                startId - width * 3 - 3 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
```

```
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild ||
                startId - width * 4 - 4 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild ||
                startId - width * 5 - 5 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 - 4}"]`).firstChild ||
                startId - width * 6 - 6 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 - 5}"]`).firstChild ||
                startId - width * 7 - 7 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 - 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 -
6}"]`).firstChild ||

                // fot left cross -- backward
                startId - width + 1 === targetId ||
                startId - width * 2 + 2 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
||
                startId - width * 3 + 3 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild ||
                startId - width * 4 + 4 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild ||
                startId - width * 5 + 5 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
```

```
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 + 4}"]`).firstChild ||
                startId - width * 6 + 6 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 + 5}"]`).firstChild ||
                startId - width * 7 + 7 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 + 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 +
6}"]`).firstChild
            ) {
                return true;
            }
            break;

        case 'rook':
            if (
                // moving straight forward
                startId + width === targetId ||
                startId + width * 2 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild ||
                startId + width * 3 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild ||
                startId + width * 4 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild ||
                startId + width * 5 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 4}"]`).firstChild ||
                startId + width * 6 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 5}"]`).firstChild ||
```

```
                startId + width * 7 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 5}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 6}"]`).firstChild ||

                // moving straight backward
                startId - width === targetId ||
                startId - width * 2 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild ||
                startId - width * 3 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild ||
                startId - width * 4 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild ||
                startId - width * 5 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 4}"]`).firstChild ||
                startId - width * 6 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 5}"]`).firstChild ||
                startId - width * 7 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 5}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 6}"]`).firstChild ||

                // moving left side straight
                startId + 1 === targetId ||
                startId + 2 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild ||
                startId + 3 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild ||
                startId + 4 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
```

```
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild ||
                startId + 5 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 4}"]`).firstChild ||
                startId + 6 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 5}"]`).firstChild ||
                startId + 7 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 5}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 6}"]`).firstChild ||

                // moving right side straight
                startId - 1 === targetId ||
                startId - 2 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild ||
                startId - 3 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild ||
                startId - 4 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild ||
                startId - 5 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 4}"]`).firstChild ||
                startId - 6 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 5}"]`).firstChild ||
                startId - 7 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 4}"]`).firstChild && !document.querySelector(`[square-
```

```
id="${startId - 5}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 6}"]`).firstChild
            ) {
                return true
            }
            break;

        case 'queen':
            if (
                // for right cross --- forward
                startId + width + 1 === targetId ||
                startId + width * 2 + 2 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
||
                startId + width * 3 + 3 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild ||
                startId + width * 4 + 4 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild ||
                startId + width * 5 + 5 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 + 4}"]`).firstChild ||
                startId + width * 6 + 6 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 + 5}"]`).firstChild ||
                startId + width * 7 + 7 === targetId &&
!document.querySelector(`[square-id = "${startId + width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 + 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 +
6}"]`).firstChild ||

                // for left cross --- forward
                startId + width - 1 === targetId ||
```

```
                startId + width * 2 - 2 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
||
                startId + width * 3 - 3 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild ||
                startId + width * 4 - 4 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild ||
                startId + width * 5 - 5 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 - 4}"]`).firstChild ||
                startId + width * 6 - 6 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 - 5}"]`).firstChild ||
                startId + width * 7 - 7 === targetId &&
!document.querySelector(`[square-id = "${startId + width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId + width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId + width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId +
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId + width * 5 - 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 -
6}"]`).firstChild ||

                // for right cross --- backward
                startId - width - 1 === targetId ||
                startId - width * 2 - 2 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
||
                startId - width * 3 - 3 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild ||
                startId - width * 4 - 4 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
```

```
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild ||
                startId - width * 5 - 5 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 - 4}"]`).firstChild ||
                startId - width * 6 - 6 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 - 5}"]`).firstChild ||
                startId - width * 7 - 7 === targetId &&
!document.querySelector(`[square-id = "${startId - width - 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 -
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 - 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 - 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 - 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 -
6}"]`).firstChild ||

                // fot left cross -- backward
                startId - width + 1 === targetId ||
                startId - width * 2 + 2 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
||
                startId - width * 3 + 3 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild ||
                startId - width * 4 + 4 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild ||
                startId - width * 5 + 5 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 + 4}"]`).firstChild ||
                startId - width * 6 + 6 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
```

```
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 + 5}"]`).firstChild ||
                startId - width * 7 + 7 === targetId &&
!document.querySelector(`[square-id = "${startId - width + 1}"]`).firstChild
&& !document.querySelector(`[square-id = "${startId - width * 2 +
2}"]`).firstChild && !document.querySelector(`[square-id = "${startId - width
* 3 + 3}"]`).firstChild && !document.querySelector(`[square-id = "${startId -
width * 4 + 4}"]`).firstChild && !document.querySelector(`[square-id =
"${startId - width * 5 + 5}"]`).firstChild &&
!document.querySelector(`[square-id = "${startId + width * 6 +
6}"]`).firstChild ||

                // moving straight forward
                startId + width === targetId ||
                startId + width * 2 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild ||
                startId + width * 3 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild ||
                startId + width * 4 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild ||
                startId + width * 5 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 4}"]`).firstChild ||
                startId + width * 6 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 5}"]`).firstChild ||
                startId + width * 7 === targetId &&
!document.querySelector(`[square-id="${startId + width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 5}"]`).firstChild &&
!document.querySelector(`[square-id="${startId + width * 6}"]`).firstChild ||

                // moving straight backward
                startId - width === targetId ||
                startId - width * 2 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild ||
```

```
                startId - width * 3 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild ||
                startId - width * 4 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild ||
                startId - width * 5 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 4}"]`).firstChild ||
                startId - width * 6 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 5}"]`).firstChild ||
                startId - width * 7 === targetId &&
!document.querySelector(`[square-id="${startId - width}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 2}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 3}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 4}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 5}"]`).firstChild &&
!document.querySelector(`[square-id="${startId - width * 6}"]`).firstChild ||

                // moving left side straight
                startId + 1 === targetId ||
                startId + 2 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild ||
                startId + 3 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild ||
                startId + 4 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild ||
                startId + 5 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 4}"]`).firstChild ||
                startId + 6 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 5}"]`).firstChild ||
```

```
                startId + 7 === targetId && !document.querySelector(`[square-
id="${startId + 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 5}"]`).firstChild && !document.querySelector(`[square-
id="${startId + 6}"]`).firstChild ||

                // moving right side straight
                startId - 1 === targetId ||
                startId - 2 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild ||
                startId - 3 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild ||
                startId - 4 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild ||
                startId - 5 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 4}"]`).firstChild ||
                startId - 6 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 5}"]`).firstChild ||
                startId - 7 === targetId && !document.querySelector(`[square-
id="${startId - 1}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 2}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 3}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 4}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 5}"]`).firstChild && !document.querySelector(`[square-
id="${startId - 6}"]`).firstChild
            ) {
                return true
            }
            break;

        case 'king':
            if (
                startId + 1 === targetId ||
                startId - 1 === targetId ||
                startId + width === targetId ||
                startId + width + 1 === targetId ||
```

```
                startId + width - 1 === targetId ||
                startId - width === targetId ||
                startId - width + 1 === targetId ||
                startId - width - 1 === targetId
            ) {
                return true
            }
            break;
        default:
            break;
    }
}


function changePlayer() {
    if (playerTurn === 'black') {
        reverseIds()
        playerTurn = 'white';
        playerDetails.textContent = 'white'
    } else {
        revertIds();
        playerTurn = 'black'
        playerDetails.textContent = 'black'
    }
}

function reverseIds() {
    const allSquares = document.querySelectorAll('#gameboard .square');
    allSquares.forEach((square, i) => {
        square.setAttribute('square-id', (width * width - 1) - i)
    })
}

function revertIds() {
    const allSquares = document.querySelectorAll('#gameboard .square');
    allSquares.forEach((square, i) => {
        square.setAttribute('square-id', i)
    })
}

function checkForWin() {
    const kings = Array.from(document.querySelectorAll('#king'));

    if (!kings.some(king => king.firstChild.classList.contains('white'))) {
        infoDisplay.innerHTML = "Black Player Wins!";
        const allSquares = document.querySelectorAll('.square');
        allSquares.forEach(square =>
square.firstChild?.setAttribute('draggable', false));
    }
```

```
    if (!kings.some(king => king.firstChild.classList.contains('black'))) {
        infoDisplay.innerHTML = "White Player Wins!";
        const allSquares = document.querySelectorAll('.square');
        allSquares.forEach(square =>
square.firstChild?.setAttribute('draggable', false));
    }
}
```

A. Program Inspection

1. In code add 1 in row, row = Math.floor((63-i)/8)+1, 2 time defended same variable using const datatype, not use try catch for handling the errors.
2. Static because there are syntax errors, which are easily identified by static testing.
3. Runtime errors like infinite loop,they are not identified just by reading the code.
4. Yes, because it resolves the error as soon as possible and prevents runtime errors.

Third Segment

GitHubLink

```
const userTab = document.querySelector("[data-userWeather]");
const searchTab = document.querySelector("[data-searchWeather]");
const userContainer = document.querySelector(".weather-container");

const grantAccessContainer = document.querySelector(".grant-location-
container");
const searchForm = document.querySelector("[data-searchForm]");
const loadingScreen = document.querySelector(".loading-container");
const userInfoContainer = document.querySelector(".user-info-container");
```

```
let oldTab = userTab;
const API_KEY = "d1845658f92b31c64bd94f06f7188c9c";
oldTab.classList.add("current-tab");
getfromSessionStorage();

function switchTab(newTab) {
    if (newTab !== oldTab) {
        oldTab.classList.remove("current-tab");
        oldTab = newTab;
        oldTab.classList.add("current-tab");

        if (!searchForm.classList.contains("active")) {
            // Show the search form if it's not active
            userInfoContainer.classList.remove("active");
            grantAccessContainer.classList.remove("active");
            searchForm.classList.add("active");
        } else {
            // Switch to user weather tab and display weather info
            searchForm.classList.remove("active");
            userInfoContainer.classList.remove("active");
            getfromSessionStorage();
        }
    }
}

userTab.addEventListener("click", () => switchTab(userTab));
searchTab.addEventListener("click", () => switchTab(searchTab));

// Check if coordinates are already present in session storage
function getfromSessionStorage() {
    const localCoordinates = sessionStorage.getItem("user-coordinates");
    if (!localCoordinates) {
        // Show grant access container if no coordinates found
        grantAccessContainer.classList.add("active");
    } else {
        const coordinates = JSON.parse(localCoordinates);
        fetchUserWeatherInfo(coordinates);
    }
}

async function fetchUserWeatherInfo(coordinates) {
    const { lat, lon } = coordinates;
    grantAccessContainer.classList.remove("active");
    loadingScreen.classList.add("active");

    // API call to fetch user weather information
    try {
        const response = await fetch(
```

```
            `https://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${
lon}&appid=${API_KEY}&units=metric`
        );

        if (!response.ok) {
            throw new Error("Network response was not ok " +
response.statusText);
        }

        const data = await response.json();
        loadingScreen.classList.remove("active");
        userInfoContainer.classList.add("active");
        renderWeatherInfo(data);
    } catch (err) {
        loadingScreen.classList.remove("active");
        // Show an error message to the user
        alert("Could not fetch weather data. Please try again later.");
        console.error("Error fetching user weather info:", err);
    }
}

function renderWeatherInfo(weatherInfo) {
    const cityName = document.querySelector("[data-cityName]");
    const countryIcon = document.querySelector("[data-countryIcon]");
    const desc = document.querySelector("[data-weatherDesc]");
    const weatherIcon = document.querySelector("[data-weatherIcon]");
    const temp = document.querySelector("[data-temp]");
    const windspeed = document.querySelector("[data-windspeed]");
    const humidity = document.querySelector("[data-humidity]");
    const cloudiness = document.querySelector("[data-cloudiness]");

    console.log(weatherInfo);

    cityName.innerText = weatherInfo?.name || "City not found";
    countryIcon.src =
`https://flagcdn.com/144x108/${weatherInfo?.sys?.country.toLowerCase()}.png`;
    desc.innerText = weatherInfo?.weather?.[0]?.description || "No description
available";
    weatherIcon.src =
`http://openweathermap.org/img/w/${weatherInfo?.weather?.[0]?.icon}.png`;
    temp.innerText = `${weatherInfo?.main?.temp} °C` || "Temperature not
available";
    windspeed.innerText = `${weatherInfo?.wind?.speed} m/s` || "Wind speed not
available";
    humidity.innerText = `${weatherInfo?.main?.humidity}%` || "Humidity not
available";
    cloudiness.innerText = `${weatherInfo?.clouds?.all}%` || "Cloudiness not
available";
```

```
}

function getLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(showPosition, showError);
    } else {
        // Show an alert if geolocation is not supported
        alert("Geolocation is not supported by this browser.");
    }
}

function showPosition(position) {
    const userCoordinates = {
        lat: position.coords.latitude,
        lon: position.coords.longitude,
    };

    sessionStorage.setItem("user-coordinates",
JSON.stringify(userCoordinates));
    fetchUserWeatherInfo(userCoordinates);
}

const grantAccessButton = document.querySelector("[data-grantAccess]");
grantAccessButton.addEventListener("click", getLocation);

const searchInput = document.querySelector("[data-searchInput]");

searchForm.addEventListener("submit", (e) => {
    e.preventDefault();
    let cityName = searchInput.value.trim();

    if (cityName === "") {
        alert("Please enter a city name.");
        return;
    } else {
        fetchSearchWeatherInfo(cityName);
    }
});

async function fetchSearchWeatherInfo(city) {
    loadingScreen.classList.add("active");
    userInfoContainer.classList.remove("active");
    grantAccessContainer.classList.remove("active");

    try {
        const response = await fetch(
            `https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=$
{API_KEY}&units=metric`
```

```
        );

        if (!response.ok) {
            throw new Error("Network response was not ok " +
response.statusText);
        }

        const data = await response.json();
        loadingScreen.classList.remove("active");
        userInfoContainer.classList.add("active");
        renderWeatherInfo(data);
    } catch (err) {
        loadingScreen.classList.remove("active");
        alert("Could not fetch weather data for the specified city. Please
check the city name and try again.");
        console.error("Error fetching search weather info:", err);
    }
}

// Function to handle geolocation errors
function showError(error) {
    switch (error.code) {
        case error.PERMISSION_DENIED:
            alert("User denied the request for Geolocation.");
            break;
        case error.POSITION_UNAVAILABLE:
            alert("Location information is unavailable.");
            break;
        case error.TIMEOUT:
            alert("The request to get user location timed out.");
            break;
        case error.UNKNOWN_ERROR:
            alert("An unknown error occurred.");
            break;
    }
}
```

A. Program Inspection

1. Potential API Errors, Error Handling for Fetch Requests, Geolocation Error Handling, User Input Validation.

2. Static because there are syntax errors, which are easily identified by static testing.
3. Runtime errors like infinite loop,they are not identified just by reading the code and Performance Issues.
4. Yes, because it resolves the error as soon as possible and prevents runtime errors.