Design Size

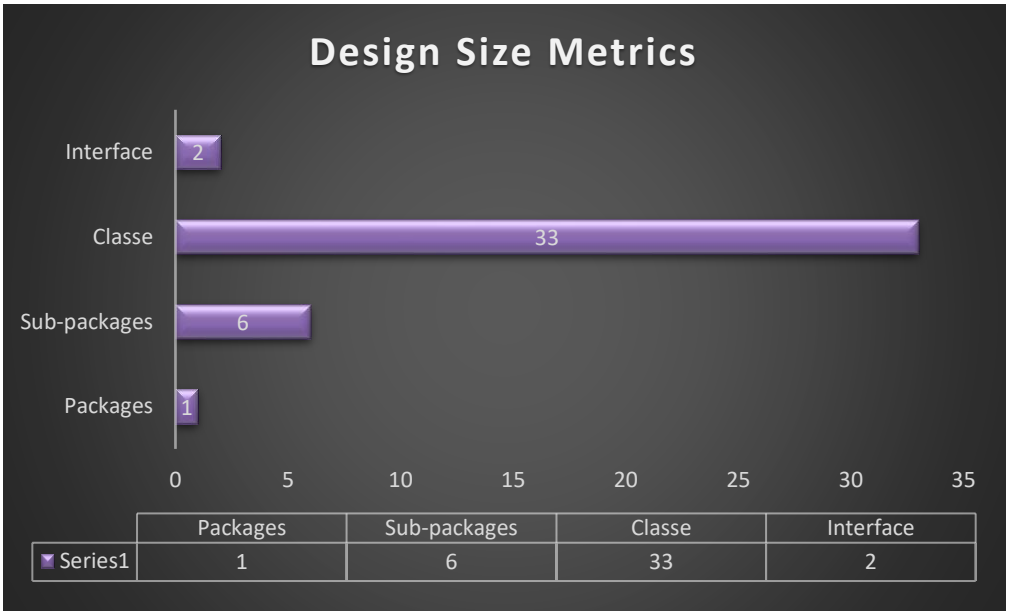
Definition: Design size measure size in terms of packages, design patterns, classes, interfaces, abstract classes, operations, and methods in a software project.

Measuring Techniques

- **Packages**: Package in Java is a mechanism to encapsulate a group of classes.

Number of Package, NoP = 1
Number of subpackages, sp = 6
Number of classes, NoC = 33
Interfaces (Java), I = 2
Abstract classes, AC = 5

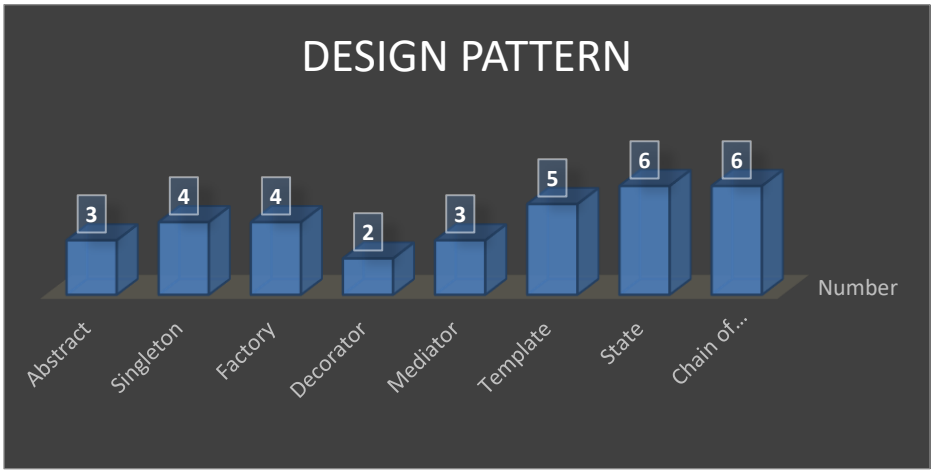
- Measure procedure: Manually + java program (Showed in Code Size file).
- Language : Java, c++



- **Design patterns:** Design patterns provide a standard terminology and are specific to particular scenario.

  - Number of design patterns, NoD = 8
  - Measure procedure: Manually.
  - Language : OOP.

Pattern	Class
Abstract	Porter_stemmer.java, HuffmanTuple.java, FileWriteHelper.java
Singleton	Huffman.java, ProcessSearchFile.java, MethodFind.java, MethodCount.java
Factory	Command.java, Similarity.java, getTfIdf.java, Node.java
Decorator	TfIdfCalculate.java, BoxAndWhiskerChart.java
Mediator	DecompressedFileWriteHelper.java, Filewriter.java, Compress.java
Template	mainEncode.java, mainDecode.java, Decompress.java, CosineSimilarity.java, LineOfCode.java
State	CloneCheck.java, FileContentReader.java, ProjectReader.java, FileCount.java, PreProcessing.java, FrequencyMapCreate.java
Chain of responsibility	Search.java, keyword.java, GrepContent.java, Filereader.java CompressedFileWriteHelper.java, Average_LOC.java



- **Public Methods:** Methods which are access by all classes and start with public keyword

Number of public methods in all classes, NoM = 57

  - Measure procedure:Programatic.
  - Language : Java, C++

```java
package software_metrics_lab_final;

import java.io.*;

class PublicMethods {

static int countOccurrences(String str, String word)
{
    // split the string by spaces in a
    String a[] = str.split(" ");

    // search for pattern in a
    int count = 0;
    for (int i = 0; i < a.length; i++)
    {
    // if match found increase count
    if (word.equals(a[i]))
        count++;
    }

    return count;
  }

public static void main(String args[])
{
    String str ="";
    String word = "public";
    System.out.println(countOccurrences(str, word));
 }
}
```

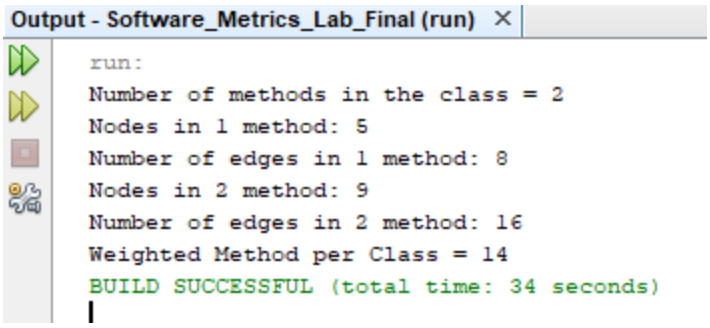
- **Methods or operations:** A method/operation is a block of code which only runs when it is called.
  - **Number of parameters, NoP = 37**
  - **Number of over loaded versions of a method or operation, OM = 5**

- Measure procedure: Manually.
- Language : OOP

- **Weighted methods per class (WMC) :** Summing the weights of the methods in a class, where weights are cyclomatic complexity factors for each method.
  - Measure procedure: Programatic.
  - Language : OOP.

#java program for wmc

```
1   package software_metrics_lab_final;
2
3   import java.util.Scanner;
4
5   public class WMC {
6       public static void main(String[] args) {
7
8           Scanner scan = new Scanner(System.in);
9
10           System.out.print("Number of methods in the class = ");
11           int x = scan.nextInt();
12
13           int wmc = 0;
14           for(int i = 1; i<=x; i++){
15           System.out.print("Nodes in "+i+" method: ");
16           int n = scan.nextInt();
17
18           System.out.print("Number of edges in "+i+" method: ");
19           int e = scan.nextInt();
20
21           int c  = e - n + 2;
22           wmc = wmc + c;
23
24           }
25
26           System.out.println("Weighted Method per Class = "+wmc);
27       }
28  }
29
```



Total WMC for the project : 137

- **Data declarations:** The number of variable initialized through the project.

  - **Data Declarations, DD = 619**

    - Measure procedure: Programatic.
    - Language : OOP

```
1   package software_metrics_lab_final;
2
3   import java.io.BufferedReader;
4   import java.io.File;
5   import java.io.FileReader;
6   import java.io.IOException;
7   import java.util.List;
8   import java.util.regex.Matcher;
9   import java.util.regex.Pattern;
    import static software_metrics_lab_final.code_size.Dlinesum;
    import static software_metrics_lab_final.code_size.fileName;
12  import static software_metrics_lab_final.code_size.listf;
    import static software_metrics_lab_final.code_size.methodsum;
14
15
16  public class Data_declaration {
17      public static String fileName;
18      public static long Dlinesum=0, methodsum=0;
19
20      public static String projectName = "C:\\Users\\User\\Desktop\\6th Semester\\LAB\\SMF\\coding_help new";
21
22      public static void main(String[] args) throws IOException {
23
24          long Dline = 0;
            String line = null;
26          long totalchar = 0;
27          int methods=0;
28
29          List<File> list=listf(projectName);
30
31          for(File file: list) {
32                  if(file.getAbsolutePath().endsWith(".java")) {
33                      fileName = file.getAbsolutePath();
```

```
                    }

                try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {
                                        while ((line = reader.readLine()) != null){
                    totalchar += line.length();

                    if (isDataDeclaration(line)) {

                                            Dline++;
                    }

                    else if(isMethod(line)) {
                            methods++;
                            }

                                            }

                }catch (IOException e) {
                                            e.printStackTrace();
                            }

                methodsum+=methods;
                Dlinesum += Dline;
                System.out.println("Total Data Declaration: " + Dlinesum);
                System.out.println("Total methods: " + methodsum);
                System.out.println("total char "+totalchar);
            }
        }
       // line = reader.readLine());
         private static boolean isDataDeclaration(String line){
                long Dline = 0;
                String[] datalist = null;
```

```
            if (line.contains("int") || line.contains("boolean") || line.contains("byte") || line.contains("char")
                        || line.contains("short") || line.contains("long") || line.contains("float") || line.c
                        || line.contains("String")) {

                return true;
            }
            return false;
        }


        private static boolean isMethod(String line) {
        Pattern pattern = Pattern.compile("(public|protected|private|static|\\s) +[\\w\\<\\>\\[\\],\\s]+\\s+(\
        Matcher matcher = pattern.matcher(line);

        return matcher.find();
    }
```

```
utput - Software_Metrics_Lab_Final (run)
run:
Total Data Declaration: 8
Total methods: 1
total char 890
Total Data Declaration: 36
Total methods: 7
total char 3627
Total Data Declaration: 97
Total methods: 20
```