Part A:

Source Project: apache-tomcat-8.5.71-src

Understande Tool
javax.el.ELClass DIT=1

DesigniteJava Tool
ELClass DIT = 0

Description: these results of CK matrices are extracted from two different tools and both show different result of same class understand tool shows Depth of Inheritance tree(DIT) = 1 and Designite Java Tool shows DIT = 0, in my point of view the value extracted from understand is accurate because javax.el.ELClass have one child class named javax.el.ELClass.ELClass, however DesigniteJava Tool generated incorrect value

Understande Tool
javax.el.StaticFieldELResolver WMC=24

DesigniteJava
StaticFieldELResolver WMC = 18

Description: these two different tools generated different values for same class but understand tool generated accurate value because javax.el.StaticFieldELResolver have 24 methods rather then 18 number of methods that generated by Designite Java tool. Changes in values are due to efficiency of tool and understand tool analyze and detect each method and class from software as compare to Designite java tool.

Understande Tool
javax.el.ValueReference LCOM = 33

DesigniteJava
ValueReference LCOM = 0.0

Description: I have selected javax.el.ValueReference class to extract value for lack of cohesion of methods and compare it's value with two different tools, as understand tool calculated LCOM = 33 for javax.el.ValueReference class, however Designite java tool calculated LCOM = 0.0 for same class. In my point of view designite java tool cucullated value accurately because LCOM must be between 0 and 1 where 0 means every method uses all instance variable and 1 means every method uses only one instance variable so Designite java tool provide accurate value for javax.el.ValueReference and it is written that normalized version range of values between 0 and 1

Compression between Understande and Designite java tool:
Understand and designite java tool are mostly similar tools but there is one main difference that designite java extract different code smells but understand tool does not provide such functionality, however understand tool is best tool to extract verity of matrices and it is consist of user friendly interface where user can easy use any functionality and it provide customization facility as compare to designite tool,

designite tool does not provide user interface where we can use different functionality and it provide fixed number of matrices with short form of matrices name.

Understand tool can be used by any user due to its interface but designite tool only used by professional person that have related knowledge because it operates on Terminal on mac so user must know appropriate commands for execution and have related information. Moreover understand tool also provide graph representation of any software it shows result in graph but designite tool does not provide this functionality.
Understand tool perform any task quickly with less computer resources but designite java use more computer resources and it take more execution time as compare to understand tool.

Refactoring Operations:
Feature Envy:
Move method (testClear, test)
Extract method (addBootModuleName, BootModule)

Long Method:
Extract method(addInitParameter,Parameter)

Long Parameter list:
introduce Parameter object(addincludeCoda,includeprelude)

```
Searching classpath folders ...
Parsing the source code ...
Resolving symbols...
Computing metrics...
Detecting code smells...
Exporting analysis results...
wrapping up ...
--Analysis summary--
        Total LOC analyzed: 399985      Number of packages: 186
        Number of classes: 3740 Number of methods: 36652
-Total architecture smell instances detected-
        Cyclic dependency: 193  God component: 37
        Ambiguous interface: 0  Feature concentration: 57
        Unstable dependency: 45 Scattered functionality: 0
        Dense structure: 1
-Total design smell instances detected-
        Imperative abstraction: 26      Multifaceted abstraction: 15
        Unnecessary abstraction: 83     Unutilized abstraction: 651
        Feature envy: 131       Deficient encapsulation: 427
        Unexploited encapsulation: 8    Broken modularization: 26
        Cyclically-dependent modularization: 115        Hub-like modularization: 8
        Insufficient modularization: 387        Broken hierarchy: 345
        Cyclic hierarchy: 12    Deep hierarchy: 0
        Missing hierarchy: 22   Multipath hierarchy: 4
        Rebellious hierarchy: 22        Wide hierarchy: 10
-Total implementation smell instances detected-
        Abstract function call from constructor: 4      Complex conditional: 693
        Complex method: 1132    Empty catch clause: 628
        Long identifier: 494    Long method: 158
        Long parameter list: 587        Long statement: 2214
        Magic number: 8014      Missing default: 228
----
Done.
```