CC ANALYZER

CC Reengineering
CC Open Source
CC Outsourcing
CC Consulting
CC Training
CC ANALYZER
CC AUDITOR
CC ARTISAN
CC ASSESS
CC OPAIRA

Test Coverage Monitor
for C, C++, COBOL, Java, JSP, and PL/I

CC ANALYZER
Test Coverage Monitor
for C, C++, COBOL, Java, JSP, and PL/I

Description:

CC ANALYZER measures the test coverage of C/C++, COBOL, Java, JSP, and PL/I programs and provides dynamic quality assurance in software development and program maintenance.

Dynamic and systematic testing increases the quality of applications. **CC ANALYZER** is instrumental in helping to avoid many of the problems which normally occur during production launches of new or altered applications.

Dialog Language	German, English
Areas of Application	Software Development
	Software Maintenance
	Quality Assurance
	Software Security
	e-Commerce
	Tuning
	Test Environment
Languages Supported	C/C++
	COBOL
	Java
	JSP
	PL/I
Platforms	Windows
	IBM z/OS
	UNIX
	AIX

You think there's no such thing as 100 % certainty? Think again. With **CC ANALYZER** you have everything you need to thoroughly test your applications before making them operational. There's no "let's see what happens" when **CC ANALYZER** is part of the application testing process. You can finally make this transition with complete security and minimal manual input.

CC ANALYZER is a test coverage monitoring tool - secure, efficient, exact and comprehensive. **CC ANALYZER** works automatically and is the perfect dynamic quality analysis tool.

CC ANALYZER for C/C++, COBOL, Java, JSP, and PL/I

- provides exact and logically structured methods for testing applications
- organizes and optimizes all information needed throughout test cycles
- provides dynamic quality assurance in software development and program maintenance

This systematic testing automatically defines and sets high quality standards which directly lead to a drastic reduction in the problems that often come up during operational introduction of new or altered applications.

The need for security and efficiency in test monitoring continues to grow exponentially to new business possibilities. New markets, e-business, e-commerce and the resulting proximity to customers demand dynamic changes in the world of IT development and production. This makes costs, time management and resources key factors in economic success.

And this redefines what it is that companies need to focus on:

- improved quality and shortening of test cycles
- objective assessment of test processes
- transparent production of test results
- documentation of reliable information on test runs

cc gmbh

CC ANALYZER
Test Coverage Monitor
for C, C++, COBOL, Java, JSP, and PL/I

Characteristics

CC ANALYZER is the quality assurance tool to measure test coverage of applications of software development and program maintenance.

Selected characteristics are:

Measure the percentage of code executed

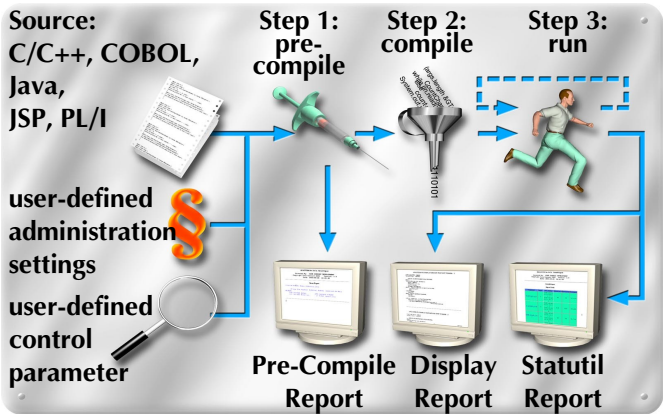
- Measure the comprehensiveness of code executed in development, test and production environments
- Show comprehensiveness of program runs by the number of code intervals in program

Identify code not executed

- Identify specific code not executed in a test run
- Count number of times each command was executed

Increase the quality and optimization of test data elements

- Provides all information required for compiling quality test data and for optimizing existing data
- Substantially increases quality of test data during development and modification/ enhancement processes



Specify test range

- Analyzes entire systems or selected individual programs and intervals
- Targeted testing of test coverage range during maintenance task

Tailor and automate code insertion

- Perform specific automated code modifications, e. g. special forms of debugging or tracing
- Automated insertion of comments or customized program measurement

Produce transparent results in HTML and text format

- Report showing the total number of intervals executed in the test and the percentage of the total of all intervals that this executed set represents

What else does **CC ANALYZER** do?

- **CC ANALYZER** for Java analyzes all Java classes (Applets, Servlets, Beans, etc.)
- Provides reliable information on the quality of test data and test processes
- Helps to optimize test data
- Trace function in targeted test runs for easy localization of errors
- Eases fast and smooth introduction to foreign programs through quality test run information with trace functionality
- Uncovers overlaps in test runs
- Shows test coverage of one or more test runs
- Eases recognition of program processes with customized test data
- Assists in reduction of unproductive test runs by producing quality analysis results
- Offers reliable test documentation with possibility to synchronize to pre-defined test end criteria
- Full integration into Eclipse

CC ANALYZER for C – HTML-Report

Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-06-30 16:16:15

Display Summary

Source Input File: count.c
Statistics File: statfile_c.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-06-30-16.07.47.00 / 2011-06-30-16.07.47.00	7	2	28.57 %
Test 2	2011-06-30-16.10.11.00 / 2011-06-30-16.10.11.00	7	6	85.71 %
TOTAL	2011-06-30-16.07.47.00 / 2011-06-30-16.10.11.00	7	7	100.00 %

CC ANALYZER for C - Report for Test Number: 1

Source Input File: count.c
Statistics File: statfile_c.dat

```
         void CountChars (FILE *ptr) {
********     char ch;
*      *     int count = 0;
                 while (!feof(ptr)) {
********         ch = fgetc(ptr);
                 if (!feof(ptr))
********            count++;
             }
********     printf("No. of chars counted = %d\n", count);
          }
          void main(int argc, char *argv[]) {
(000001)     FILE *fptr;
             char filename[100];
             if (argc > 1) {
********       strcpy(filename, argv[1]);
*      *       fptr = fopen(filename, "r");
*      *       CountChars(fptr);      fclose(fptr);
             }
             else
(000001)        printf("Enter the file name as arg to count chars\n");
          }
```

Out of 7 Intervals, 2 (28.57%) were executed
1 Test Run(s) represented from 2011-06-30 16.07.47.00
to 2011-06-30 16.07.47.00

CC ANALYZER for C - Report for Test Number: 2

Source Input File: count.c
Statistics File: statfile_c.dat

```
         void CountChars (FILE *ptr) {
(000001)    char ch;
            int count = 0;
               while (!feof(ptr)) {
(000722)       ch = fgetc(ptr);
               if (!feof(ptr))
(000721)          count++;
            }
(000001)    printf("No. of chars counted = %d\n", count);
         }
         void main(int argc, char *argv[]) {
(000001)    FILE *fptr;
            char filename[100];
            if (argc > 1) {
(000001)      strcpy(filename, argv[1]);
              fptr = fopen(filename, "r");
              CountChars(fptr);      fclose(fptr);
            }
            else
********       printf("Enter the file name as arg to count chars\n");
         }
```

```
Out of    7 Intervals,     6 ( 85.71%) were executed
     1 Test Run(s) represented from 2011-06-30 16.10.11.00
                                 to 2011-06-30 16.10.11.00
```

Accumulated Results of Test Runs

Source Input File: count.c
Statistics File: statfile_c.dat

```
         void CountChars (FILE *ptr) {
(000001)    char ch;
            int count = 0;
              while (!feof(ptr)) {
(000722)       ch = fgetc(ptr);
               if (!feof(ptr))
(000721)          count++;
            }
(000001)    printf("No. of chars counted = %d\n", count);
         }
         void main(int argc, char *argv[]) {
(000002)    FILE *fptr;
            char filename[100];
            if (argc > 1) {
(000001)       strcpy(filename, argv[1]);
               fptr = fopen(filename, "r");
               CountChars(fptr);      fclose(fptr);
            }
            else
(000001)       printf("Enter the file name as arg to count chars\n");
         }
```

```
Out of    7 Intervals,     7 (100.00%) were executed
     2 Test Run(s) represented from 2011-06-30 16.07.47.00
                                 to 2011-06-30 16.10.11.00
```

CC ANALYZER for COBOL – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-07-22 10:03:33

Display Summary

Source Input File: DEMO1.COB
Statistics File: statfile_cobol.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-07-22 10.02.52.65 / 2011-07-22 10.03.01.18	25	14	56.00 %
Test 2	2011-07-22 10.03.06.06 / 2011-07-22 10.03.28.87	25	22	88.00 %
TOTAL	2011-07-22 10.02.52.65 / 2011-07-22 10.03.28.87	25	25	100.00 %

CC ANALYZER for COBOL - Report for Test Number: 1

Source Input File: DEMO1.COB
Statistics File: statfile_cobol.dat

```
        003050*
        003051 PARA-2.
(000001)003052     DISPLAY 'Enter Text to display in subprogram :'
        003053     ACCEPT PARM
        003054     CALL CALL-DEMO2 USING PARM.
        003055
        003100
        003200 PARA-2-X.
(000001)003300     EXIT.
        003400*
        003500 PARA-3.
********003600     MOVE SPACE TO PARM
*       *003700     CALL CALL-DEMO2 USING PARM.
        003800 PARA-3-X.
********003900     EXIT.
*       *004000*
        004100 PARA-4.
********004200     DISPLAY 'http://www.caseconsult.com'.
        004300 PARA-4-X.
```

```
Out of   25 Intervals,     14 ( 56.00% ) were Executed
        329 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.02.52.65
                                 to 2011-07-22 10.03.01.18
```

```
                CC ANALYZER for COBOL - Report for Test Number: 2

Source Input File:   DEMO1.COB
Statistics File:     statfile_cobol.dat

--------------------------------------------------------------------------------

         003050*
         003051 PARA-2.
********003052     DISPLAY 'Enter Text to display in subprogram :'
*       *003053     ACCEPT PARM
*       *003054     CALL CALL-DEMO2 USING PARM.
*       *003055
*       *003100
         003200 PARA-2-X.
********003300     EXIT.
*       *003400*
         003500 PARA-3.
(000001)003600     MOVE SPACE TO PARM
         003700     CALL CALL-DEMO2 USING PARM.
         003800 PARA-3-X.
(000001)003900     EXIT.
         004000*
         004100 PARA-4.
(000001)004200     DISPLAY 'http://www.caseconsult.com'.
         004300 PARA-4-X.

--------------------------------------------------------------------------------

Out of   25 Intervals,     22 ( 88.00% ) were Executed
         329 Line(s) Read from Source File
       1 Test Run(s) Represented from 2011-07-22 10.03.06.06
                                   to 2011-07-22 10.03.28.87
```

```
                      Accumulated Results of Test Runs

Source Input File:   DEMO1.COB
Statistics File:     statfile_cobol.dat

--------------------------------------------------------------------------------

         003050*
         003051 PARA-2.
(000001)003052     DISPLAY 'Enter Text to display in subprogram :'
         003053     ACCEPT PARM
         003054     CALL CALL-DEMO2 USING PARM.
         003055
         003100
         003200 PARA-2-X.
(000001)003300     EXIT.
         003400*
         003500 PARA-3.
(000001)003600     MOVE SPACE TO PARM
         003700     CALL CALL-DEMO2 USING PARM.
         003800 PARA-3-X.
(000001)003900     EXIT.
         004000*
         004100 PARA-4.
(000001)004200     DISPLAY 'http://www.caseconsult.com'.
         004300 PARA-4-X.

--------------------------------------------------------------------------------

Out of   25 Intervals,     25 ( 100.00% ) were Executed
         329 Line(s) Read from Source File
           4 Record(s) Read from Statistics File
           2 Statistics Record(s) Pertain to This Program
       2 Test Run(s) Represented from 2011-07-22 10.02.52.65
                                   to 2011-07-22 10.03.28.87
```

```
                    CC ANALYZER for COBOL – STATISTICS-Report

Licenced by:  CASE CONSULT DEVELOPMENT

(C) CC GmbH 2011                                Version: 5.4
Date: 2011-06-30  16:16:15


  PROGRAM:  DEMO2

               RUN DATE         INTERVALS EXECUTED PERCENT INSTRUMENTATION DATE
          ---------------------- --------- -------- ------- ---------------------
          2011-06-30(13:48:17)         5         4     80%   2011-06-30(11:52:50)
          2011-06-30(13:53:36)         5         2     40%   2011-06-30(11:52:50)


------------------------------- REPORT TOTALS ---------------------------------

  PROGRAM    RUNS        RUN DATE          INTERVALS   EXECUTED   PERCENT
  --------  -----  ---------------------  ---------  ---------  ---------
  DEMO2         2        2011-06-30              5          5       100%
  --------  -----                          ---------  ---------  ---------
  TOTAL:        2                                5          5       100%

101 - REPORT COMPLETED
```

```
          CC ANALYZER for COBOL - DISPLAY-REPORT FOR TEST NUMBER:   1

SOURCE INPUT FILE:   DEMO2
STATISTICS FILE:     STATFILE


                PROCEDURE DIVISION USING PARM1, PARM2.
                PARA-1.
(000001)            MOVE 'DEMO2' TO PARM1
                    MOVE 'Y' TO MAIN-ENTRY-SW
                    MOVE +1234 TO PARM2.
                    IF PARM2 = 9999
(000001)                DISPLAY 'PARM2 HAS AN IMPOSSIBLE VALUE'
                        GO TO PARA-EXIT
                    END-IF
********            EXIT PROGRAM.
                PARA-EXIT.
(000001)            DISPLAY 'ABORTING PROGRAM'.
                PARA-EXIT-X.
(000001)            EXIT PROGRAM.

OUT OF    5 INTERVALS,    4 ( 80%) WERE EXECUTED.
         89 RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(13:48:17.35)
                                     TO 2011-06-30(13:48:18.89)
```

```
        CC ANALYZER for COBOL - DISPLAY-REPORT FOR TEST NUMBER:   2

SOURCE INPUT FILE:   DEMO2
STATISTICS FILE:     STATFILE


                PROCEDURE DIVISION USING PARM1, PARM2.
                PARA-1.
(000010)            MOVE 'DEMO2' TO PARM1
                    MOVE 'Y' TO MAIN-ENTRY-SW
                    MOVE +1234 TO PARM2.
                    IF PARM2 = 9999
********                DISPLAY 'PARM2 HAS AN IMPOSSIBLE VALUE'
*      *                GO TO PARA-EXIT
                    END-IF
(000010)            EXIT PROGRAM.
                PARA-EXIT.
********            DISPLAY 'ABORTING PROGRAM'.
*      *        PARA-EXIT-X.
********            EXIT PROGRAM.

OUT OF    5     INTERVALS,    2 ( 40%) WERE EXECUTED.
         89     RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(13:53:36.05)
                                      TO 2011-03-30(13:53:37.49)
```

```
          CC ANALYZER for COBOL - ACCUMULATED RESULTS OF TEST RUNS

 SOURCE INPUT FILE:   DEMO2
 STATISTICS FILE:     STATFILE


                 PROCEDURE DIVISION USING PARM1, PARM2.
                 PARA-1.
 (000011)            MOVE 'DEMO2' TO PARM1
                     MOVE 'Y' TO MAIN-ENTRY-SW
                     MOVE +1234 TO PARM2.
                     IF PARM2 = 9999
 (000001)                DISPLAY 'PARM2 HAS AN IMPOSSIBLE VALUE'
                         GO TO PARA-EXIT
                     END-IF
 (000010)            EXIT PROGRAM.
                 PARA-EXIT.
 (000001)            DISPLAY 'ABORTING PROGRAM'.
                 PARA-EXIT-X.
 (000001)            EXIT PROGRAM.

 OUT OF    5    INTERVALS,    5 (100%) WERE EXECUTED.
          89    RECORDS READ FROM SOURCE FILE
           2 TEST RUN(S) REPRESENTED FROM 2011-06-30(13:48:17.35)
                                       TO 2011-06-30(13:53:37.49)
```

CC ANALYZER for C++ – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-07-22 10:06:58

Display Summary

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

Runs	Run Date	Intervals	Executed	Percent
Test_1	2011-07-22 10.06.23.00 / 2011-07-22 10.06.38.00	16	12	75.00 %
Test_2	2011-07-22 10.06.42.00 / 2011-07-22 10.06.55.00	16	12	75.00 %
TOTAL	2011-07-22 10.06.23.00 / 2011-07-22 10.06.55.00	16	14	87.55 %

CC ANALYZER for C++ - Report for Test Number: 1

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

```
          if (a <= 10)                                  // IF without brace
********     cout << "Number is less than 10";
*      *                                                // End of IF without brace
          else if ((a >= 10)
                   && (a <= 100))                       // ELSE IF without brace
********     cout << "Number is between 10 and 100";
*      *                                                // End of ELSE IF without brace
          else                                          // ELSE without brace
(000001)     cout << "Number is greater than 100";
                                                        // ELSE without brace
(000001)
          if (a <= 50)                                  // IF with brace
          {
********     cout << "\nNumber is less than 50";
          }                                             /* End of IF with brace */
          else if ((a >= FIFTY)
                   && (a <= 100))                       /* ELSE IF with brace */
          {
********     cout << "\nNumber is between 50 and 100";
          }                                             /* End of ELSE IF with brace */
          else                                          // ELSE with brace
```

```
Out of   16 Intervals,     12 ( 75.00% ) were Executed
       288 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.06.23.00
                                 to 2011-07-22 10.06.38.00
```

CC ANALYZER for C++ - Report for Test Number: 2

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

```
           if (a <= 10)                                    // IF without brace
********      cout << "Number is less than 10";
*      *                                                   // End of IF without brace
           else if ((a >= 10)
                    && (a <= 100))                         // ELSE IF without brace
(000001)      cout << "Number is between 10 and 100";
                                                           // End of ELSE IF without brace
           else                                            // ELSE without brace
********      cout << "Number is greater than 100";
*      *                                                   // ELSE without brace
(000001)
           if (a <= 50)                                    // IF with brace
           {
********      cout << "\nNumber is less than 50";
           }                                               /* End of IF with brace */
           else if ((a >= FIFTY)
                    && (a <= 100))                         /* ELSE IF with brace */
           {
(000001)      cout << "\nNumber is between 50 and 100";
           }                                               /* End of ELSE IF with brace */
           else                                            // ELSE with brace
```

```
Out of   16 Intervals,     12 ( 75.00% ) were Executed
       288 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.06.42.00
                                to 2011-07-22 10.06.55.00
```

Accumulated Results of Test Runs

Source Input File: sample.cpp
Statistics File: statfile_cpp.dat

```
           if (a <= 10)                                    // IF without brace
********      cout << "Number is less than 10";
*      *                                                   // End of IF without brace
           else if ((a >= 10)
                    && (a <= 100))                         // ELSE IF without brace
(000001)      cout << "Number is between 10 and 100";
                                                           // End of ELSE IF without brace
           else                                            // ELSE without brace
(000001)      cout << "Number is greater than 100";
                                                           // ELSE without brace
(000002)
           if (a <= 50)                                    // IF with brace
           {
********      cout << "\nNumber is less than 50";
           }                                               /* End of IF with brace */
           else if ((a >= FIFTY)
                    && (a <= 100))                         /* ELSE IF with brace */
           {
(000001)      cout << "\nNumber is between 50 and 100";
           }                                               /* End of ELSE IF with brace */
           else                                            // ELSE with brace
```

```
Out of   16 Intervals,     12 ( 75.00% ) were Executed
       288 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 10.06.42.00
                                to 2011-07-22 10.06.55.00
```

CC ANALYZER for Java – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-06-30 16:16:15

Display Summary

Runs	Run Date	Intervals	Executed	Percent
Test_1	2011-06-30-15.06.36.890000 / 2011-06-30-15.06.37.380000	6	2	33.00 %
Test_2	2011-06-30-15.07.12.750000 / 2011-06-30-15.07.13.140000	6	5	83.00 %
TOTAL	2011-06-30-15.06.36.890000 / 2011-06-30-15.07.13.140000	6	6	100.00 %

CC ANALYZER for Java - Report for Test Number: 1

SOURCE INPUT FILE: count.java
STATISTICS FILE: statfile_java.dat

```
         import java.io.*;
         public class Count {
            public static void CountChars(Reader in) throws IOException {
********      int count = 0;
*      *      while (in.read() != -1)
********         count++;
*      *
********      System.out.println("No. of chars counted = " + count);
*      *
            }
            public static void main(String[] args) throws Exception {
(000001)       if (args.length >= 1)
********       CountChars(new FileReader(args[0]));
*      *       else
*      *
(000001)          System.err.println("Enter the file name as arg to count chars");
            }
         }
```

```
OUT OF    6 Intervals,     2 ( 33%) WERE EXECUTED
         66 Record(s) Read from Source File
     1 Test Run(s) Represented from 2011-06-30-15.06.36.890000
                                 to 2011-06-30-15.06.37.380000
```

CC ANALYZER for Java - Report for Test Number: 2

SOURCE INPUT FILE: count.java
STATISTICS FILE: statfile_java.dat

```
         import java.io.*;
         public class Count {
            public static void CountChars(Reader in) throws IOException {
(000001)       int count = 0;
               while (in.read() != -1)
(000447)          count++;
(000001)       System.out.println("No. of chars counted = " + count);
            }
            public static void main(String[] args) throws Exception {
(000001)       if (args.length >= 1)
(000001)          CountChars(new FileReader(args[0]));
               else
********          System.err.println("Enter the file name as arg to count chars");
*       *
            }
         }
```

```
OUT OF    6 Intervals,     5 ( 83%) were executed
         66 Record(s) read from Source File
     1 Test Run(s) represented from 2011-06-30-15.07.12.750000
                                 to 2011-06-30-15.07.13.140000
```

Accumulated Results of Test Runs

SOURCE INPUT FILE: count.java
STATISTICS FILE: statfile_java.dat

```
         import java.io.*;
         public class Count {
            public static void CountChars(Reader in) throws IOException {
(000001)       int count = 0;
               while (in.read() != -1)
(000447)          count++;
(000001)       System.out.println("No. of chars counted = " + count);
            }
            public static void main(String[] args) throws Exception {
(000002)       if (args.length >= 1)
(000001)          CountChars(new FileReader(args[0]));
               else
(000001)          System.err.println("Enter the file name as arg to count chars");
            }
         }
```

```
OUT OF    6 Intervals,     6 (100%) were executed
         66 Record(s) read from Source File
     2 Test Run(s) represented from 2011-06-30-15.06.36.890000
                                 to 2011-06-30-15.07.13.140000
```

CC ANALYZER for JSP – HTML-Report
Licenced by: CASE CONSULT DEVELOPMENT
(C) CC GmbH 2011 Version: 1.3
Date: 2011-07-22 09:15:37

Display Summary

Source Input File: numguess.jsp
Statistics File: statfile_jsp.dat

Runs	Run Date	Intervals	Executed	Percent
Test 1	2011-07-22 09.12.55.40 / 2011-07-22 09.12.55.45	4	2	50.00 %
Test 2	2011-07-22 09.12.55.40 / 2011-07-22 09.13.29.25	4	3	75.00 %
TOTAL	2011-07-22 09.12.55.40 / 2011-07-22 09.13.29.25	4	4	100.00 %

CC ANALYZER for JSP - Report for Test Number: 1

Source Input File: numguess.jsp
Statistics File: statfile_jsp.dat

```
          if (numguess.getSuccess()) {
********
                                        %>
          Congratulations!  You got it.
          And after just <%= numguess.getNumGuesses() %> tries.<p>
          <%
********     numguess.reset();
                                  %>
          Care to <a href="numguess.jsp">try again</a>?
        <%
          } else if (numguess.getNumGuesses() == 0) {
(000001)
                                                        %>
          Welcome to the Number Guess game.<p>

          I'm thinking of a number between 1 and 100.<p>

          <form method=get>
          What's your guess? <input type=text name=guess>
          <input type=submit value="Submit">
          </form>
        <%
          } else {
(000002)
```

```
Out of    4 Intervals,      2 ( 50.00% ) were Executed
       105 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 09.12.55.40
                                 to 2011-07-22 09.12.55.45
```

```
                ANALYZER for JSP - Report for Test Number:   2

Source Input File:   numguess.jsp
Statistics File:     statfile_jsp.dat
-------------------------------------------------------------------------------
             if (numguess.getSuccess()) {
(000001)
                                          %>
          Congratulations!  You got it.
          And after just <%= numguess.getNumGuesses() %> tries.<p>
          <%
(000001)     numguess.reset();
                                  %>
          Care to <a href="numguess.jsp">try again</a>?
        <%
           } else if (numguess.getNumGuesses() == 0) {
********
                                                          %>
          Welcome to the Number Guess game.<p>

          I'm thinking of a number between 1 and 100.<p>
          <form method=get>
          What's your guess? <input type=text name=guess>
          <input type=submit value="Submit">
          </form>
        <%
           } else {
(000009)

-------------------------------------------------------------------------------
Out of    4 Intervals,      3 ( 75.00% ) were Executed
        105 Line(s) Read from Source File
     1 Test Run(s) Represented from 2011-07-22 09.12.55.40
                                    to 2011-07-22 09.13.29.25
```

```
                Accumulated Results of Test Runs

Source Input File:   numguess.jsp
Statistics File:     statfile_jsp.dat
-------------------------------------------------------------------------------
             if (numguess.getSuccess()) {
(000001)
                                          %>
          Congratulations!  You got it.
          And after just <%= numguess.getNumGuesses() %> tries.<p>
          <%
(000001)     numguess.reset();
                                  %>
          Care to <a href="numguess.jsp">try again</a>?
        <%
           } else if (numguess.getNumGuesses() == 0) {
(000001)
                                                          %>
          Welcome to the Number Guess game.<p>
          I'm thinking of a number between 1 and 100.<p>
          <form method=get>
          What's your guess? <input type=text name=guess>
          <input type=submit value="Submit">
          </form>
        <%
           } else {
(000011)

-------------------------------------------------------------------------------
Out of    4 Intervals,      4 ( 100.00% ) were Executed
        105 Line(s) Read from Source File
          4 Record(s) Read from Statistics File
          2 Statistics Record(s) Pertain to This Program
     2 Test Run(s) Represented from 2011-07-22 09.12.55.40
                                    to 2011-07-22 09.13.29.25
```

CC ANALYZER for PL/I Output Results
(C) CC GmbH 2011

```
                    CC ANALYZER for PL/I – STATISTICS-REPORT

LICENSED BY:  CASE CONSULT DEVELOPMENT


  (C) CC GmbH 2011                                    VERSION: 5.4
  DATE: 2011-06-30  16:12:47


  PROGRAM:  DEMOP2

               RUN DATE        INTERVALS EXECUTED PERCENT INSTRUMENTATION DATE
---------------------- --------- -------- ------- --------------------
         2011-06-30(15:03:35)        4        3      75%   2011-06-30(14:26:09)
         2011-06-30(15:07:23)        4        2      50%   2011-06-30(14:26:09)


------------------------------- REPORT TOTALS --------------------------------

  PROGRAM    RUNS        RUN DATE         INTERVALS   EXECUTED   PERCENT
  --------  -----  ---------------------  ---------  ---------  ---------
  DEMOP2        2        2011-06-30             4          4       100%
  --------  -----                          ---------  ---------  ---------
  TOTAL:        2                               4          4       100%

101 - REPORT COMPLETED
```

```
          CC ANALYZER for PL/I - DISPLAY-REPORT FOR TEST NUMBER:   1

SOURCE INPUT FILE:   DEMOP2
STATISTICS FILE:     STATFILE


          DEMOP2: PROCEDURE(PARM1,PARM2) REORDER;
(000001)
  NON-PROCEDURAL CODE SUPPRESSED
          PARM1 = 'DEMOP1';
          MAIN_ENTRY_SW = '1'B;
          DISPLAY('PARM2');
          IF PARM2 = 9999 THEN DO;
(000001)      DISPLAY('PARM2 HAS AN IMPOSSIBLE VALUE');
              GOTO PARA_EXIT;
              END;
********  RETURN;
          PARA_EXIT:
(000001)  DISPLAY('ABORTING PROGRAM');
          RETURN;
          END DEMOP2;

OUT OF    4 INTERVALS,     3 ( 75%) WERE EXECUTED.
         59 RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(15:03:35.17)
                                      TO 2011-06-30(15:03:35.25)
```

```
          CC ANALYZER for PL/I - DISPLAY-REPORT FOR TEST NUMBER:   2

SOURCE INPUT FILE:   DEMOP2
STATISTICS FILE:     STATFILE


          DEMOP2: PROCEDURE(PARM1,PARM2) REORDER;
(000010)
  NON-PROCEDURAL CODE SUPPRESSED
          PARM1 = 'DEMOP1';
          MAIN_ENTRY_SW = '1'B;
          DISPLAY('PARM2');
          IF PARM2 = 9999 THEN DO;
********      DISPLAY('PARM2 HAS AN IMPOSSIBLE VALUE');
*      *      GOTO PARA_EXIT;
*      *      END;
(000010)  RETURN;
          PARA_EXIT:
********  DISPLAY('ABORTING PROGRAM');
*      *  RETURN;
*      *  END DEMOP2;

OUT OF    4 INTERVALS,     2 ( 50%) WERE EXECUTED.
         59 RECORDS READ FROM SOURCE FILE
          1 TEST RUN(S) REPRESENTED FROM 2011-06-30(15:07:23.62)
                                      TO 2011-06-30(15:07:23.78)
```

```
            CC ANALYZER for PL/I - ACCUMULATED RESULTS OF TEST RUNS

SOURCE INPUT FILE:   DEMOP2
STATISTICS FILE:     STATFILE


          DEMOP2: PROCEDURE(PARM1,PARM2) REORDER;
(000011)
  NON-PROCEDURAL CODE SUPPRESSED
          PARM1 = 'DEMOP1';
          MAIN_ENTRY_SW = '1'B;
          DISPLAY('PARM2');
          IF PARM2 = 9999 THEN DO;
(000001)      DISPLAY('PARM2 HAS AN IMPOSSIBLE VALUE');
              GOTO PARA_EXIT;
              END;
(000011)  RETURN;
          PARA_EXIT:
(000001)  DISPLAY('ABORTING PROGRAM');
          RETURN;
          END DEMOP2;

OUT OF    4 INTERVALS,     4 (100%) WERE EXECUTED.
         59 RECORDS READ FROM SOURCE FILE
          2 TEST RUN(S) REPRESENTED FROM 2011-06-30(15:03:35.17)
                                      TO 2011-06-30(15:07:23.78)
```

CC Germany
CC GmbH
Kreuzberger Ring 36
65205 Wiesbaden
Phone +49-611-942040
info-europe@cc-gmbh.de

CC USA
Case Consult Corporation
East Main Street 176, Suite 5
Westborough, MA 01581
Phone +1-800-836-7772
info-usa@cc-gmbh.de

www.cc-gmbh.de