Defect Programmer Assignment

Capgemini - Sprint 1
Group 3

Agenda

1. Project Idea
2. Agenda
3. Data Flow
4. Program Flow
5. Testing (CUnit , Valgrind, GCoverage)

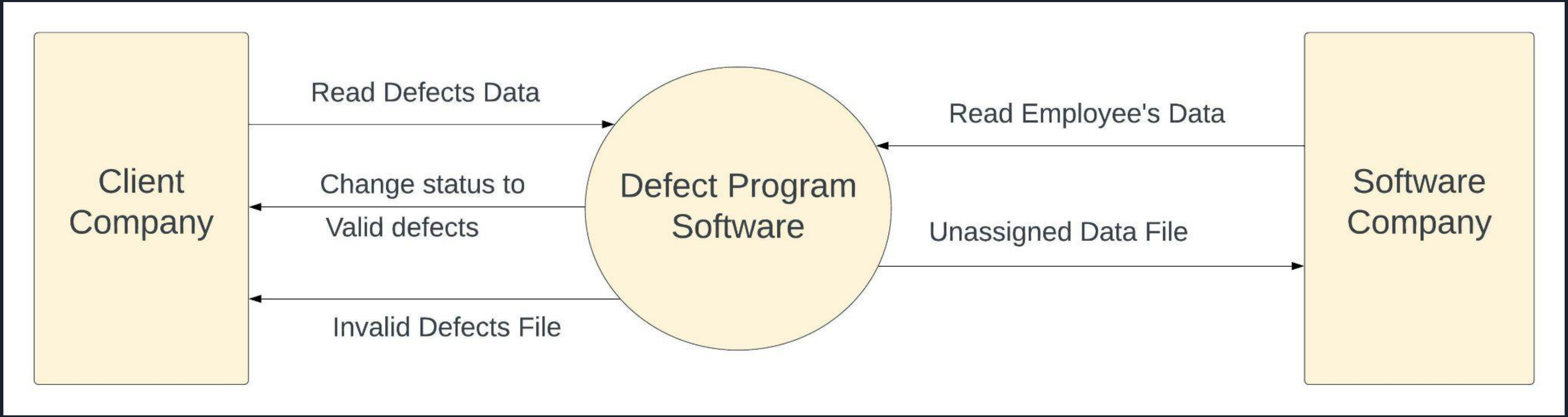
The Project Idea

To develop a software that automatically assigns the defects reported by the client company to programmers depending on the functional area they are handling.

…..

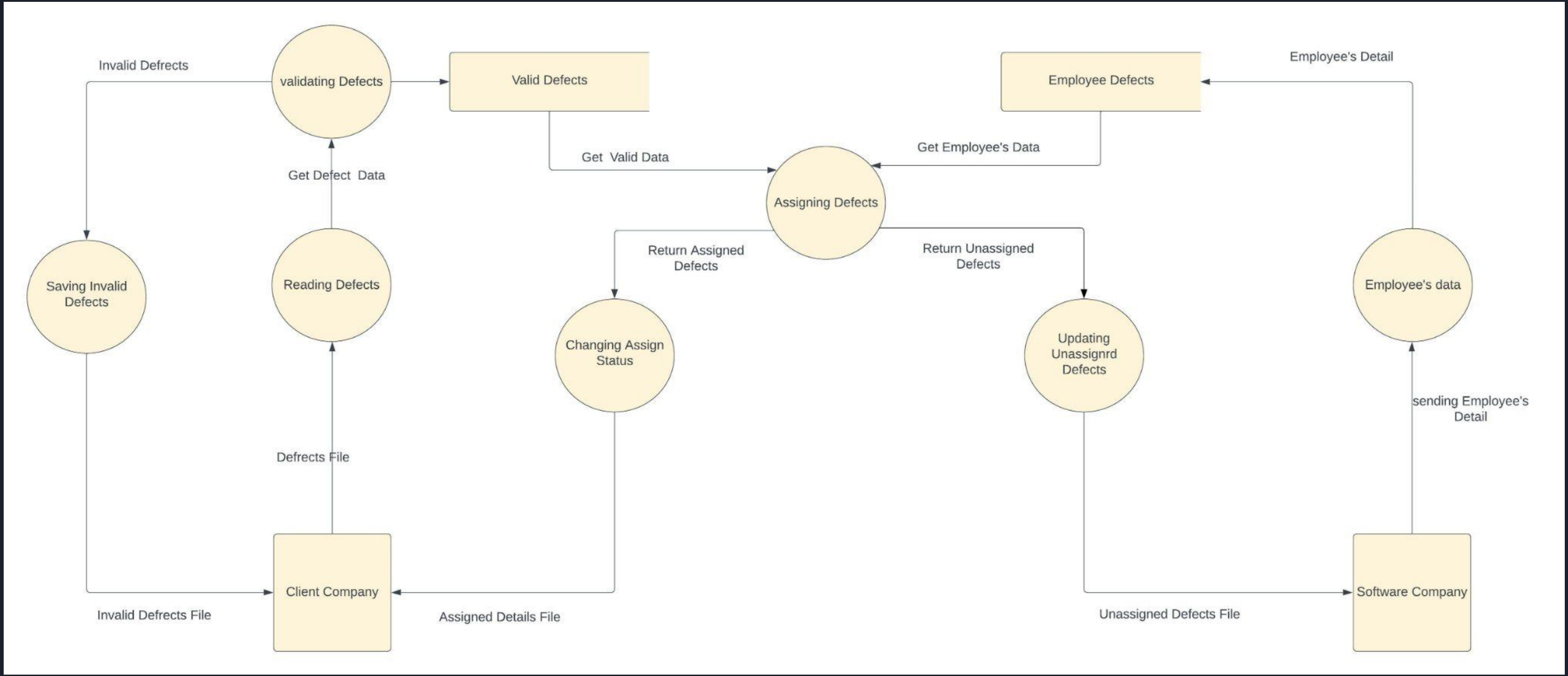
Data Flow Diagram

Level 0 DFD



Data Flow Diagram

Level 1 DFD



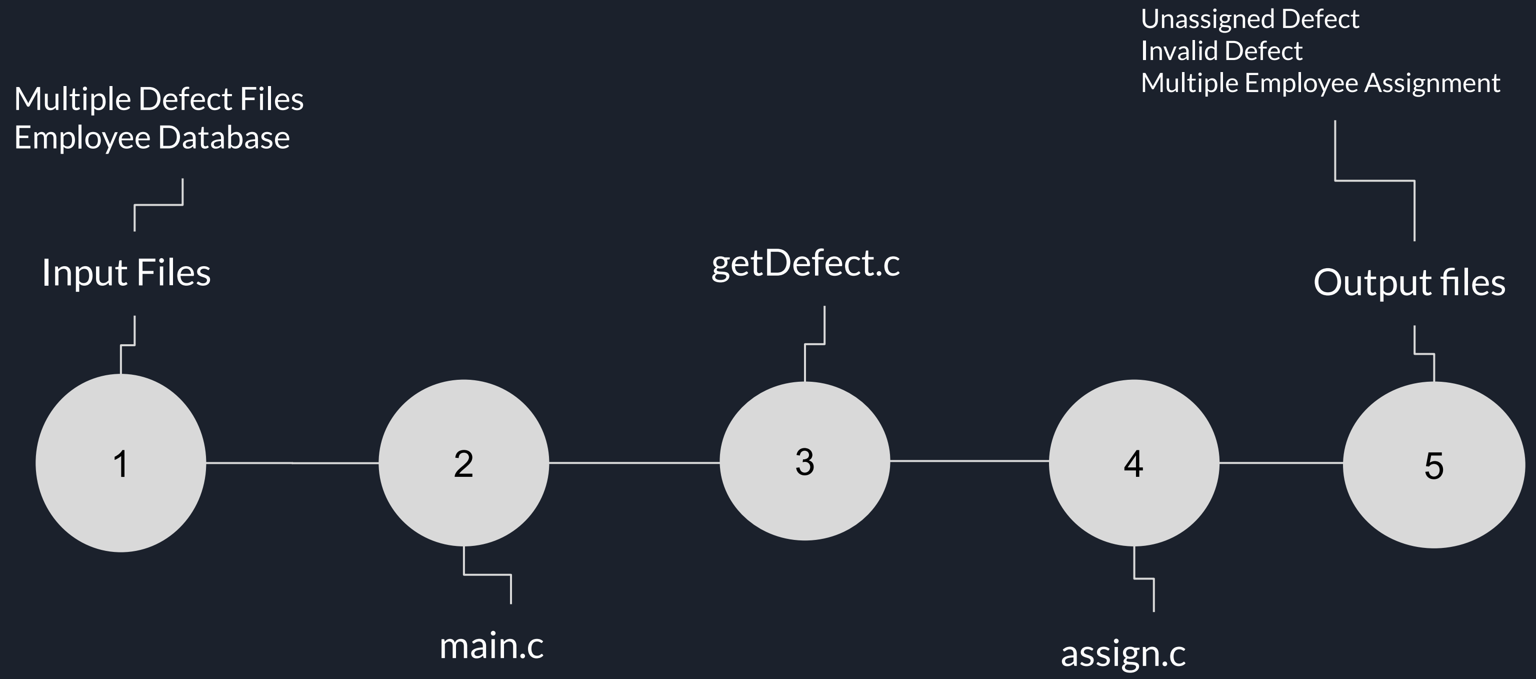
Solution

Multi Threaded

Multi File

Modular

Program Flow



main.c

1. main()

1. Input Defect files are taken as command line arguments and also validates them
2. Separate threads are created for each input files and these files are passed to getDefect() function .
3. It calls getEmployee() function to fetch data from Employee database.
4. Finally it waits for all threads to complete their work.

getEmployee()

It opens “employee.txt” database file.

Now it reads the file line by line, each line contains information of one employee.

It stores this information inside Employee Structure.

Displays error if file can't be opened for any reason.

getdefect.c

Functions

1. Get Defect

   Read the defects from the input and call assignEmployee() for valid defects

2. Check Validity

   Return true for valid and false for Invalid defects

3. Valid defect

   Store the valid defect in defect structure

4. Invalid Defect

   Store the invalid data into invalidDefect.txt

1.GetDefect()

1. It reads the defect file from input file
2. After reading it call checkvalidity()
3. If true call validDefect() Else call invalidDefect()
4. It calls assignEmployee() for valid defect

2.CheckValidity()

1. it classify the defect data into valid or invalid type.
2. It tokenizes the defect data using strtok()
3. It increment the value of count for each attribute.
4. If count is equal to 7 it returns true
5. Else it returns false

3. ValidDefect()

1. If CheckValidity() return True the defect data is Valid defect
2. It tokenizes the string using strtok()
3. Dynamically allocates memory
4. Store valid defects into their respective attribute in defect structure

4. InvalidDefect()

1. If CheckValidity() return false the defect data is invalid
2. It display invalid defect message with defect id
3. Append the invalid defect data into invalidDefect.txt

Assign.c

Functions

1.assignEmployee()- Checks for defects with status as open.
2.unassignedDefect()- Copies all unassigned Defect into separate text file.
3.createEmployeeFile()- Creates separate files for each programmer who have at least one defect assigned to him
4.searchProgrammer()- Searches for programmer suitable of open defect.

1.assignEmployee()

1. It loops through all defects and checks their status.
2. If status is open then it calls searchProgrammer() Function.
3. Defects with any other status are ignored.

2.unassignedDefect()

1. Now it opens “**uassignedDefect.txt**” file and appends all information of current defect to the last line of the file.
2. If file is not present it creates a new one.
3. Displays proper error if there is any issue with opening or writing inside this file.

3.createEmployeeFile()

1. Creates separate file for each employee, if not present already, who have at least one defect assigned to them.
2. Filename:**-<EmpID> _assignments.txt**
3. Appends employee and defect information into the file.

4.SearchProgrammer():

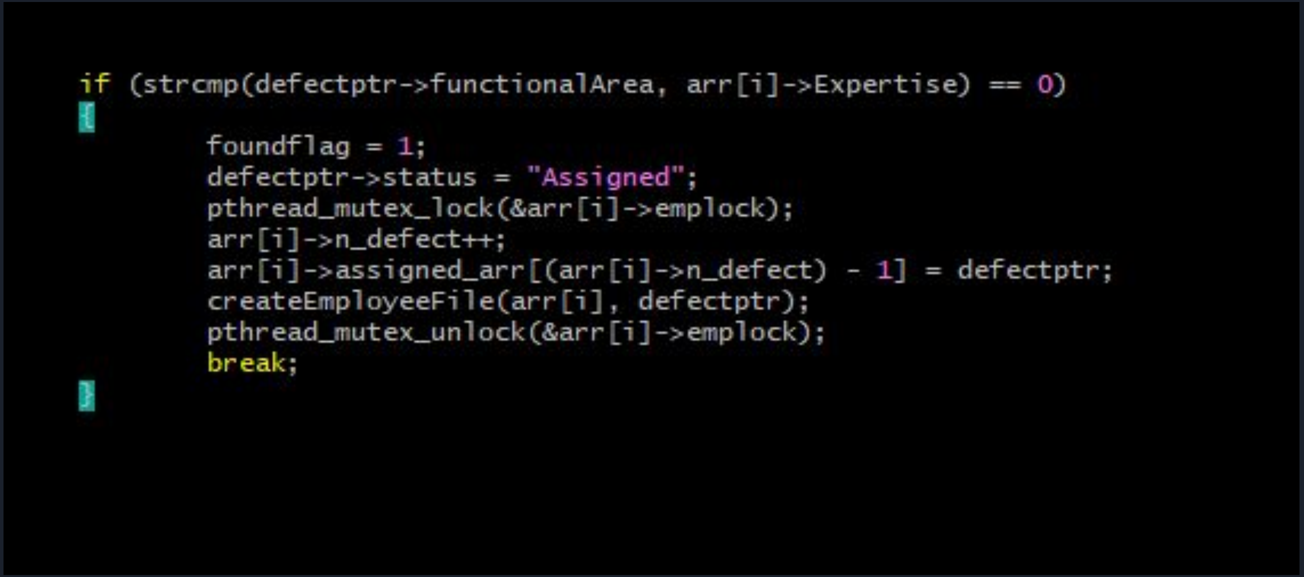
1. For each defect (passed from assignProgrammer()), searches the array for a suitable programmer to assign the defect.

2. Condition for search: Functional area of defect should match with the expertise of the programmer.

3. If a programmer is found then a mutex lock is created on the particular employee and createEmployeeFile() is called.

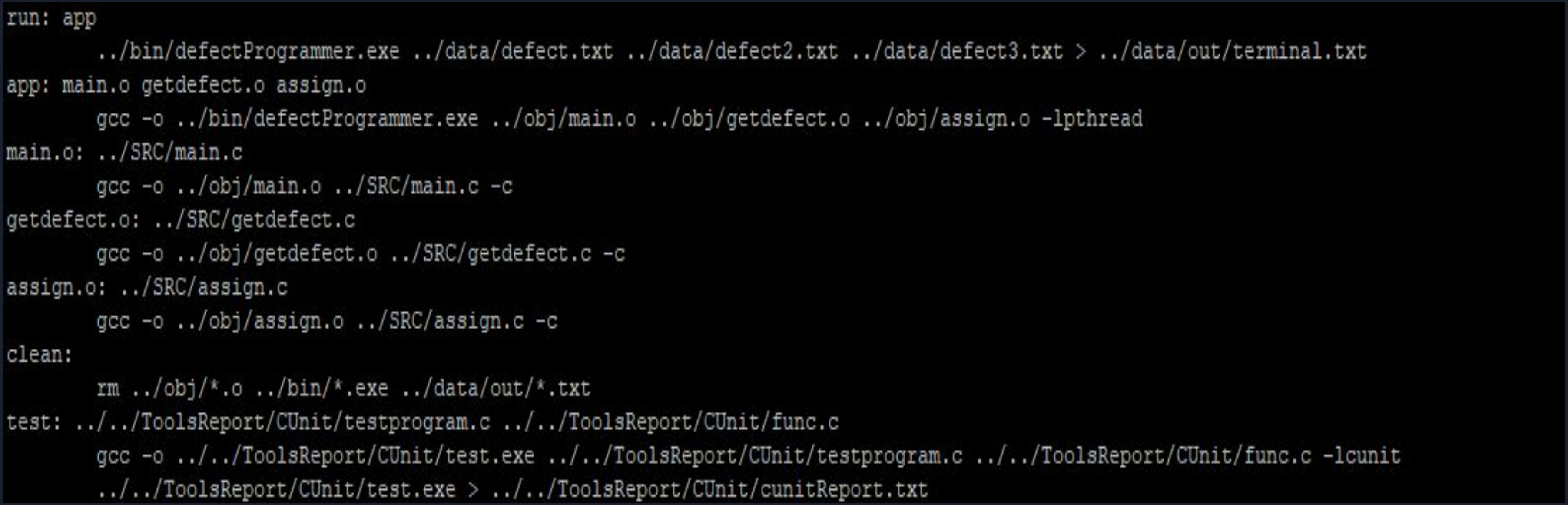
4. If no programmer is found then unassignedDefect() is called.

Mutex Lock:



```
if (strcmp(defectptr->functionalArea, arr[i]->Expertise) == 0)
{
        foundflag = 1;
        defectptr->status = "Assigned";
        pthread_mutex_lock(&arr[i]->emplock);
        arr[i]->n_defect++;
        arr[i]->assigned_arr[(arr[i]->n_defect) - 1] = defectptr;
        createEmployeeFile(arr[i], defectptr);
        pthread_mutex_unlock(&arr[i]->emplock);
        break;
}
```

MakeFile



```
run: app
	../bin/defectProgrammer.exe ../data/defect.txt ../data/defect2.txt ../data/defect3.txt > ../data/out/terminal.txt
app: main.o getdefect.o assign.o
	gcc -o ../bin/defectProgrammer.exe ../obj/main.o ../obj/getdefect.o ../obj/assign.o -lpthread
main.o: ../SRC/main.c
	gcc -o ../obj/main.o ../SRC/main.c -c
getdefect.o: ../SRC/getdefect.c
	gcc -o ../obj/getdefect.o ../SRC/getdefect.c -c
assign.o: ../SRC/assign.c
	gcc -o ../obj/assign.o ../SRC/assign.c -c
clean:
	rm ../obj/*.o ../bin/*.exe ../data/out/*.txt
test: ../../ToolsReport/CUnit/testprogram.c ../../ToolsReport/CUnit/func.c
	gcc -o ../../ToolsReport/CUnit/test.exe ../../ToolsReport/CUnit/testprogram.c ../../ToolsReport/CUnit/func.c -lcunit
	../../ToolsReport/CUnit/test.exe > ../../ToolsReport/CUnit/cunitReport.txt
```

Testing

Unit testing and Integration testing

Different Types of Test Cases Covered

File is Empty or not Opening or Invalid file type

No Programmer is Found for Defect

Wrong Format of Defects or Employee in File

Less than or More than Actual Defect Attributes

Less than or More than Actual Employee Attributes

Test Suites

Sunny Test Cases:

```
"F001:Column values in BOM reports are incorrect:Aircraft design:BOM report:21/08/2022:open:fatal"
```

```
"N001:BOM report columns not alligned properly:Aircraft design:BOM report:21/08/2022:open:niceToHave"
```

```
"F002:Unit prices are not shown while preparing invoice:Invoices:Display products:23/04/2022:close:fatal"
```

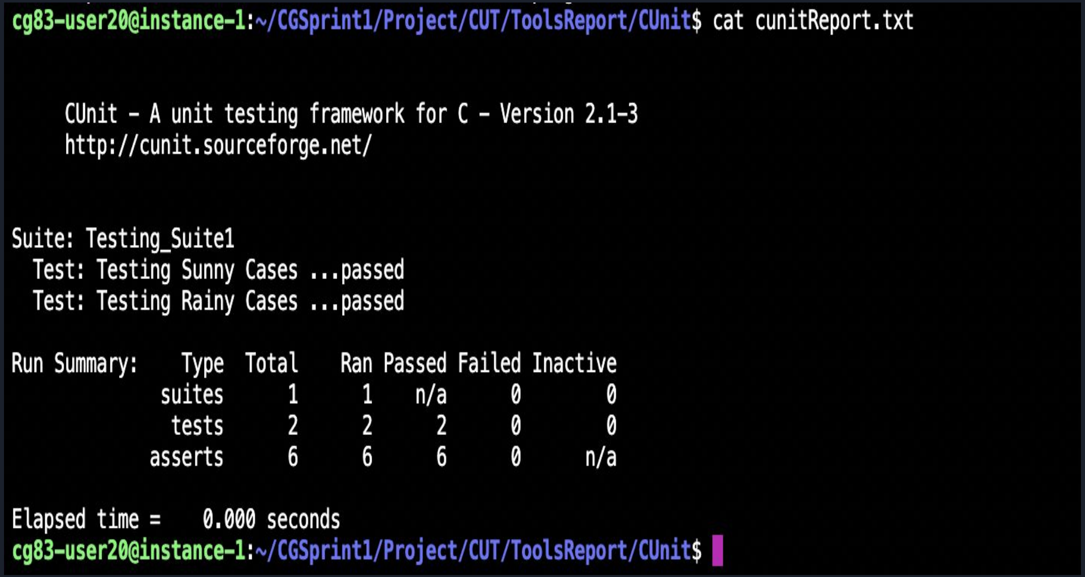
Rainy Test Cases:

```
"ID01: : : ::open:"
```

```
"L001:Aircraft:BOM report:14/09/2022:open:niceToHave"
```

```
"R096:cliant dashboard are not shown:Aircraft design:dashboard"
```

Unit testing for CheckValidity Function.



```
cg83-user20@instance-1:~/CGSprint1/Project/CUT/ToolsReport/CUnit$ cat cunitReport.txt


     CUnit - A unit testing framework for C - Version 2.1-3
     http://cunit.sourceforge.net/


Suite: Testing_Suite1
  Test: Testing Sunny Cases ...passed
  Test: Testing Rainy Cases ...passed

Run Summary:    Type  Total    Ran Passed Failed Inactive
              suites      1      1    n/a      0        0
               tests      2      2      2      0        0
             asserts      6      6      6      0      n/a

Elapsed time =    0.000 seconds
cg83-user20@instance-1:~/CGSprint1/Project/CUT/ToolsReport/CUnit$ 
```

checkValidity()

It takes 1 argument that is: String pointer

It divides the string into tokens using strtok() and increment count for each.

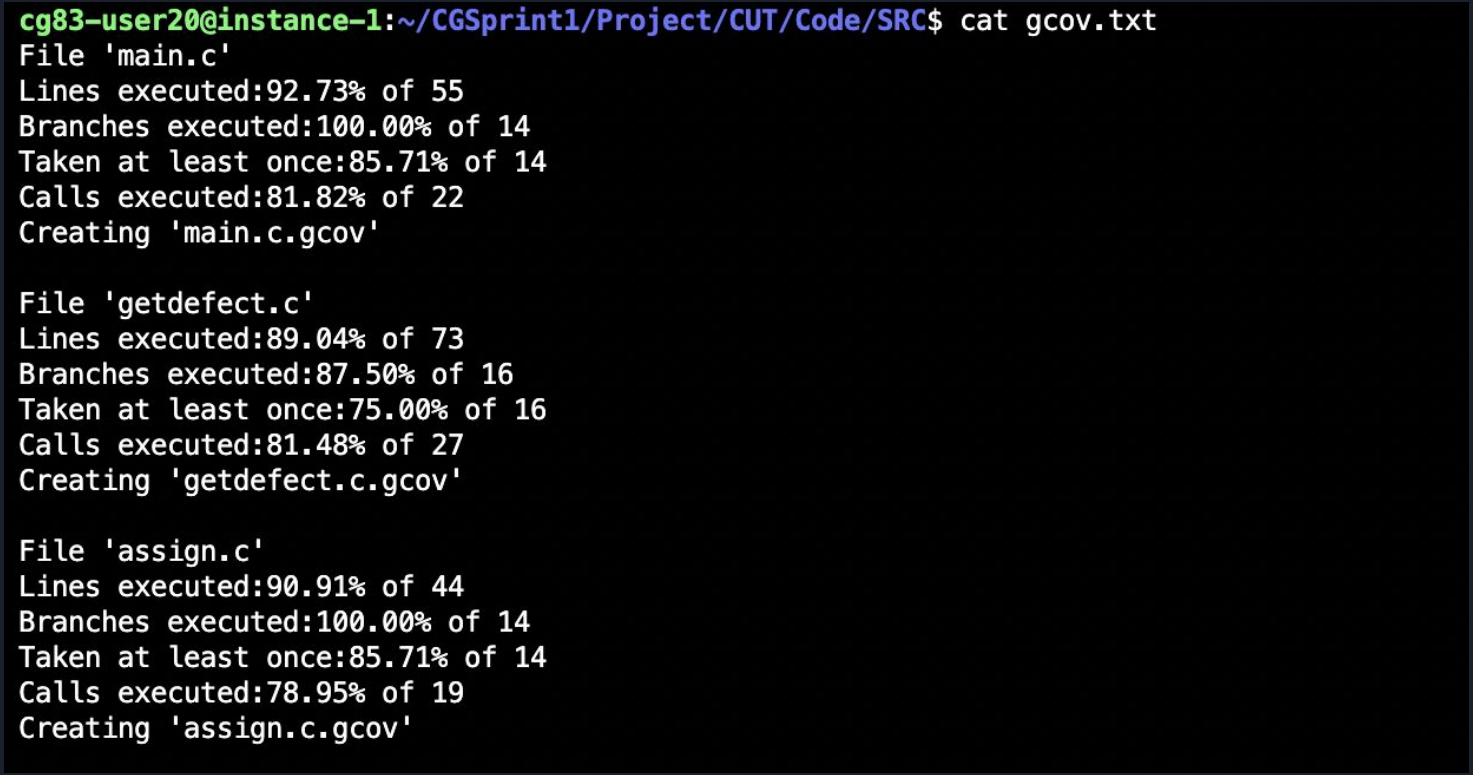
If count is equal to 7 it returns True

Else it returns False.

Valgrind Report

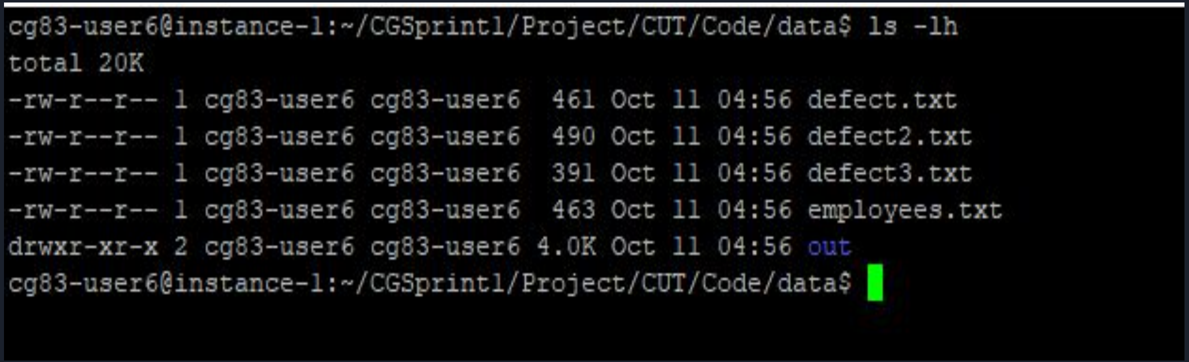
```
==80999== Memcheck, a memory error detector
==80999== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==80999== Using Valgrind-3.16.1 and LibVEX; rerun with -h for copyright info
==80999== Command: make app
==80999== Parent PID: 80998
==80999==
==80999==
==80999== HEAP SUMMARY:
==80999==     in use at exit: 147,994 bytes in 1,246 blocks
==80999==   total heap usage: 2,072 allocs, 826 frees, 407,261 bytes allocated
==80999==
==80999== LEAK SUMMARY:
==80999==    definitely lost: 0 bytes in 0 blocks
==80999==    indirectly lost: 0 bytes in 0 blocks
==80999==      possibly lost: 0 bytes in 0 blocks
==80999==    still reachable: 147,994 bytes in 1,246 blocks
==80999==         suppressed: 0 bytes in 0 blocks
==80999== Reachable blocks (those to which a pointer was found) are not shown.
==80999== To see them, rerun with: --leak-check=full --show-leak-kinds=all
==80999==
==80999== For lists of detected and suppressed errors, rerun with: -s
==80999== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

GCOV:



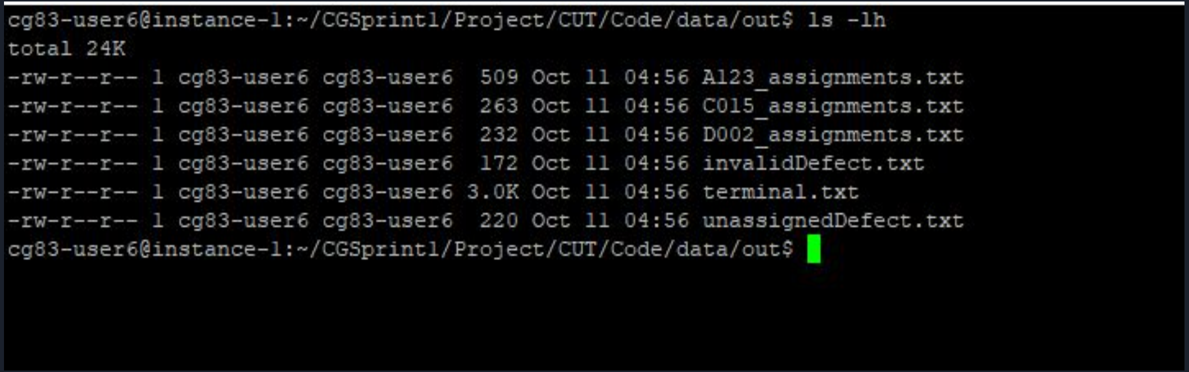
```
cg83-user20@instance-1:~/CGSprint1/Project/CUT/Code/SRC$ cat gcov.txt
File 'main.c'
Lines executed:92.73% of 55
Branches executed:100.00% of 14
Taken at least once:85.71% of 14
Calls executed:81.82% of 22
Creating 'main.c.gcov'

File 'getdefect.c'
Lines executed:89.04% of 73
Branches executed:87.50% of 16
Taken at least once:75.00% of 16
Calls executed:81.48% of 27
Creating 'getdefect.c.gcov'

File 'assign.c'
Lines executed:90.91% of 44
Branches executed:100.00% of 14
Taken at least once:85.71% of 14
Calls executed:78.95% of 19
Creating 'assign.c.gcov'
```

Integration Testing



```
cg83-user6@instance-1:~/CGSprint1/Project/CUT/Code/data$ ls -lh
total 20K
-rw-r--r-- 1 cg83-user6 cg83-user6  461 Oct 11 04:56 defect.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  490 Oct 11 04:56 defect2.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  391 Oct 11 04:56 defect3.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  463 Oct 11 04:56 employees.txt
drwxr-xr-x 2 cg83-user6 cg83-user6 4.0K Oct 11 04:56 out
cg83-user6@instance-1:~/CGSprint1/Project/CUT/Code/data$
```

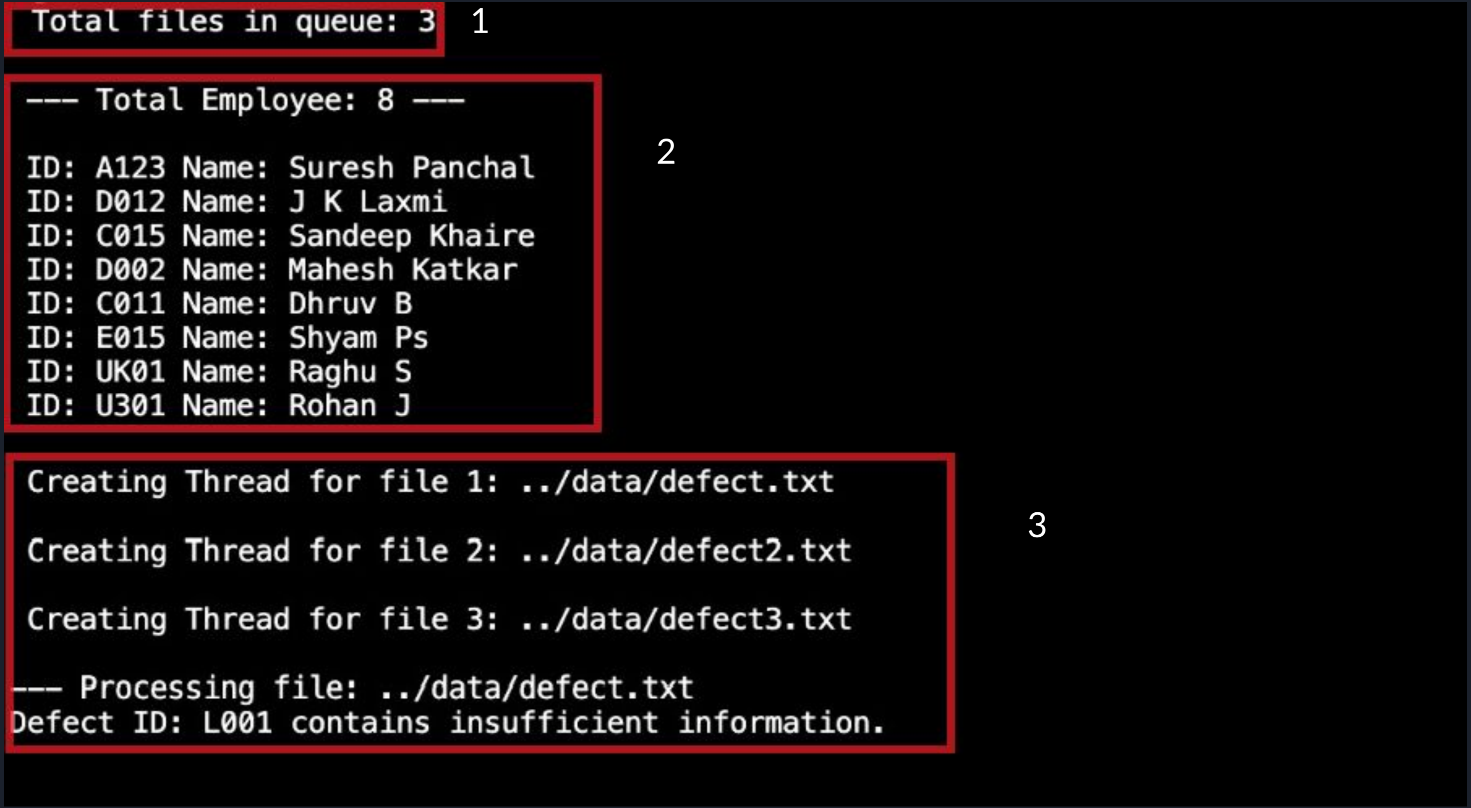
Input Files



```
cg83-user6@instance-1:~/CGSprint1/Project/CUT/Code/data/out$ ls -lh
total 24K
-rw-r--r-- 1 cg83-user6 cg83-user6  509 Oct 11 04:56 A123_assignments.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  263 Oct 11 04:56 C015_assignments.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  232 Oct 11 04:56 D002_assignments.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  172 Oct 11 04:56 invalidDefect.txt
-rw-r--r-- 1 cg83-user6 cg83-user6 3.0K Oct 11 04:56 terminal.txt
-rw-r--r-- 1 cg83-user6 cg83-user6  220 Oct 11 04:56 unassignedDefect.txt
cg83-user6@instance-1:~/CGSprint1/Project/CUT/Code/data/out$
```

Output Files

Contd..



```
 Total files in queue: 3
```

1

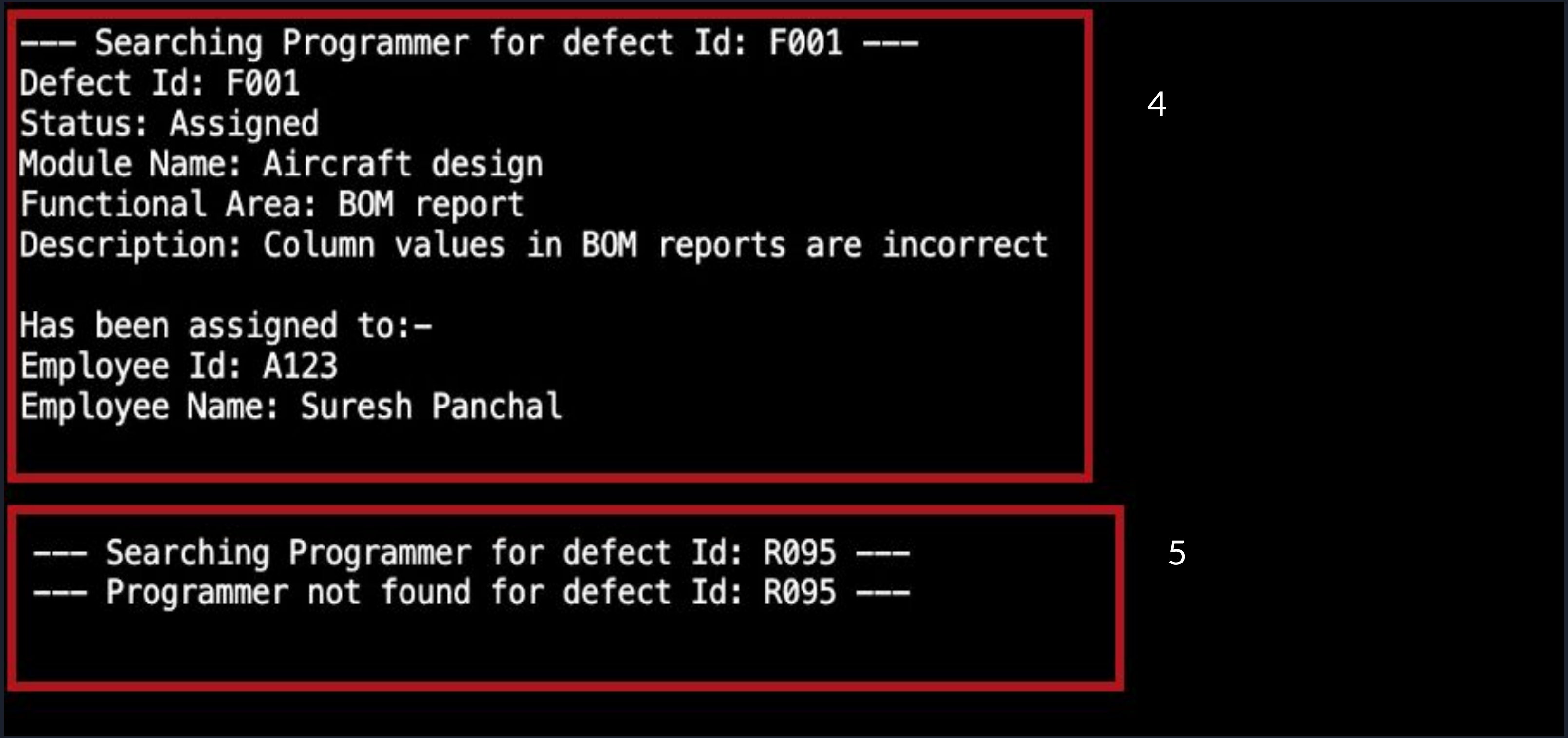
```
--- Total Employee: 8 ---

ID: A123 Name: Suresh Panchal
ID: D012 Name: J K Laxmi
ID: C015 Name: Sandeep Khaire
ID: D002 Name: Mahesh Katkar
ID: C011 Name: Dhruv B
ID: E015 Name: Shyam Ps
ID: UK01 Name: Raghu S
ID: U301 Name: Rohan J
```

2

```
 Creating Thread for file 1: ../data/defect.txt

 Creating Thread for file 2: ../data/defect2.txt

 Creating Thread for file 3: ../data/defect3.txt

--- Processing file: ../data/defect.txt
Defect ID: L001 contains insufficient information.
```

3

Contd..



```
--- Searching Programmer for defect Id: F001 ---
Defect Id: F001
Status: Assigned
Module Name: Aircraft design
Functional Area: BOM report
Description: Column values in BOM reports are incorrect

Has been assigned to:-
Employee Id: A123
Employee Name: Suresh Panchal
```

4

```
--- Searching Programmer for defect Id: R095 ---
--- Programmer not found for defect Id: R095 ---
```

5

Team CG83-Group 3

Sushant Kumar Singh

Egumamidi Sandeep Reddy

Gokul Krishna

Gundra Harshith reddy

Yuvraj C gadad

THANKS

Do you have any question?

Visit: https://github.com/HarshithReddy15/CGSprint1