GCOV

Készítette: Sztarek Norbert

Bevezetés

Teszkészítési hatékonyság

Bizalom növelése

Fejlesztési folyamat hatékonysága

Mi is a kódbefedés?

Fontossága

Tesztek alaposssága és kitejedtsége

Nem garancia

Típusai

Sor kódbefedés (Line coverage)

Elágazási kódbefedés (Branch coverage)

Ág kódbefedés (Condition coverage)

Funkcionális kódbefedés (Function coverage)

Előnyök

Jobb minőségű kód

Bizalmat adó tesztlefedettség

Hatékonyabb hibajavítás

Tesztelési hatékonyság növelése

Kihívások

Túlzott függés a kódbefedésre

"Fehér foltok" a tesztelésben

Tesztelési erőforrások

Hamis biztonságérzet

Hogyan érjük el?

Tesztelési tervezés

Tesztautomatizálás

Tesztelési keretrendszerek használata

Tesztpiramis alkalmazása

Tesztkorpusz bővítése

Refaktorálás és egyszerűsítés

GCOV

Tesztlefedetségi program

Program nem tesztelt részeit felfedi

GPROF

Profiling

Teljesítmény ellenőrzése

Modul optimalizálás

Új tesztesetek

Optimalizálás mellőzése

```
if (a != b)
  c = 1;
else
  c = 0;
```

```
      100:   12:if (a != b)
      100:   13:  c = 1;
      100:   14:else
      100:   15:  c = 0;
```

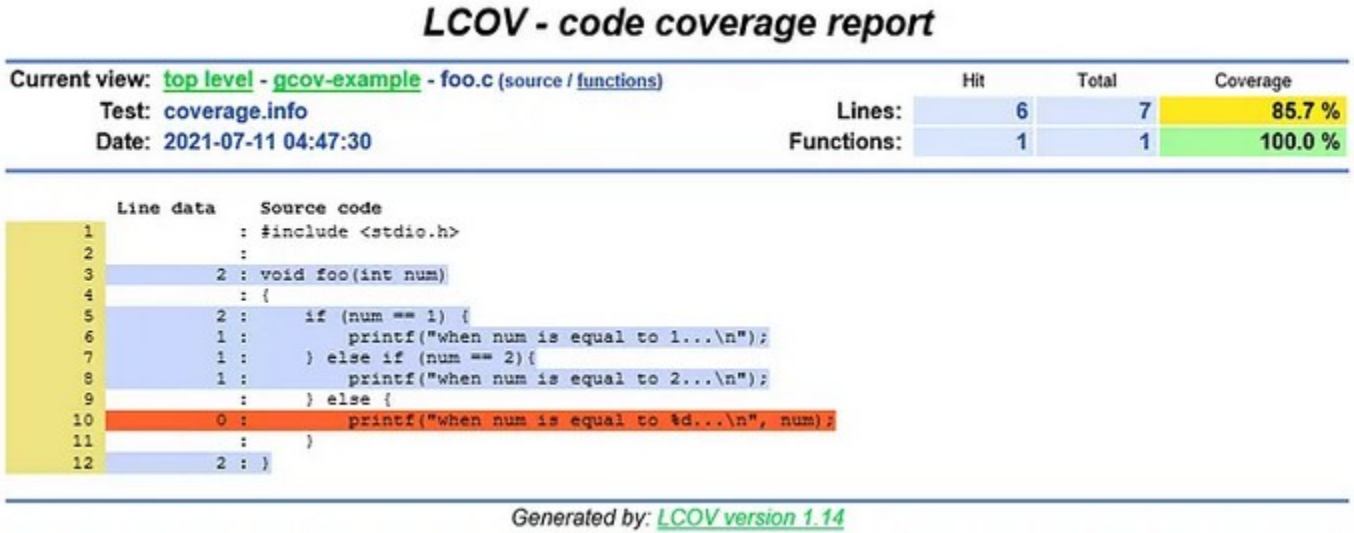
Működés

sourcefile.gcov

gprof

Csak gcc

lcov



LCOV - code coverage report

		Hit	Total	Coverage
Current view: top level - gcov-example - foo.c (source / functions)	Lines:	6	7	85.7 %
Test: coverage.info	Functions:	1	1	100.0 %
Date: 2021-07-11 04:47:30				

Line	Data	Source code
1	:	#include <stdio.h>
2	:	
3	2	: void foo(int num)
4	:	: {
5	2	: if (num == 1) {
6	1	: printf("when num is equal to 1...\n");
7	1	: } else if (num == 2){
8	1	: printf("when num is equal to 2...\n");
9	:	: } else {
10	0	: printf("when num is equal to %d...\n", num);
11	:	: }
12	2	: }

Generated by: LCOV version 1.14

Használat

gcc -fPIC -fprofile-arcs -ftest-coverage -c -Wall -Werror foo.c

gcc -fPIC -fprofile-arcs -ftest-coverage -o foo foo.o

gcov foo.c

```
  -:    0:Source:PerfectNumber.c
  -:    0:Graph:PerfectNumber.gcno
  -:    0:Data:PerfectNumber.gcda
  -:    0:Runs:2
  -:    1:#include<stdio.h>
  2:    2:int main ()
  -:    3:{
  2:    4:    int n,i,sum=0;
  2:    5:    printf("enter n\n");
  2:    6:    scanf("%d",&n);
 75:    7:    for (i=1;i<n;i++)
  -:    8:    {
 73:    9:        if (n%i==0)
  6:   10:        sum=sum+i;
  -:   11:    }
  2:   12:    if (n==sum)
  1:   13:    printf("Entered no. is a perfect no. ");
  -:   14:    else
  1:   15:    printf("Entered no. is not a perfect no.");
  2:   16:    return 0;
  -:   17:    }
```

```
  -:    0:Source:Addition.c
  -:    0:Graph:Addition.gcno
  -:    0:Data:Addition.gcda
  -:    0:Runs:1
  -:    1:#include<stdio.h>
  1:    2:void main()
  -:    3:{ int a,b,c;
  1:    4:printf("Enter two Numbers : ");
  1:    5:scanf("%d %d",&a,&b);
  1:    6:c=a+b;
  1:    7:printf("The Sum is %d",c);
  1:    8:}
```