Check Unit Test Tutorial

Relevant Link

- Check
  https://libcheck.github.io/check/
- Check Tutorial
  https://www.ccs.neu.edu/home/skotthe/classes/cs5600/fall/2015/labs/intro-check.html
- Source code
  https://github.com/libcheck/check/releases

Check

Check merupakan salah satu unit test untuk bahasa C. Unit Testing ini dikhususkan untuk OS Linux. Unit testing digunakan untuk melakukan pengecekan program secara modular (dapat digunakan untuk mengecek apakah fungsi dalam sebuah program berjalan dengan benar).

Instalasi

Debian / Linux
OS ini lebih disarankan karena merupakan yang paling didukung oleh check. Instalasi dapat dilakukan dengan langsung mengetikkan pada terminal:

```
$ sudo apt-get install check
```

Windows
Dapat menggunakan Cygwin atau MinGW dengan mendownload source yang dibuat lalu menjalankan perintah sebagai berikut ke direktori yang sudah di-unpack menggunakan terminal Cygwin atau MSYS untuk menjalankan perintah sebagai berikut:

```
$ ./configure
$ make
$ make check
$ make install
```

Selain itu, dapat juga dilakukan dengan menginstall WSL (*Windows Subsystem for Linux*) lalu menjalankan perintah yang sama dengan perintah yang dijalankan di Linux.

MacOS
Mirip dengan Debian / Linux, Instalasi dapat dilakukan dengan mengetikkan pada terminal:
Pastikan homebrew sudah ada, lalu:

```
$ brew install check
```

Pembuatan Testing

Hal yang perlu diperhatikan

Contoh file testing dan tutorial dapat dilihat di atas. Namun, dalam proses pelaksanaannya ada beberapa hal yang perlu diperhatikan, yaitu:

- Instalasi *gcovr*
  Dibutuhkan jika tidak ingin mengubah banyak dari *Makefile* yang sudah disediakan oleh Check.
- Mengubah Makefile
  Terdapat salah satu line dari Makefile yang perlu diganti yaitu dengan mengubah TST_LIBS menjadi:
  `TST_LIBS = -lcheck -lm -lpthread -lrt -lsubunit`

Pembuatan File Test

Jika ingin menggunakan Makefile yang sudah disediakan oleh Check, maka disarankan untuk mengikuti susunan *directory* yang sudah ditentukan, yaitu:

```
Money
├── Makefile
├── src
│   ├── money.c
│   └── money.h
└── tests
    └── check_money.c
```

File header dan implementasi dari header tersebut dimasukkan ke dalam folder src, lalu file tes dibuat di dalam folder tests.
Hal yang perlu diperhatikan dalam membuat file tes (*check_<nama_file>*) adalah sebagai berikut:

- **Test**
  Ditandai dengan method `START_TEST(<nama_method_test>)`. Dalam method ini, dipastikan hanya melakukan pengecekan dari 1 kasus saja menggunakan beberapa assert..
- **Test Case**
  Gabungan beberapa test. Dibuat dengan menambahkan `tcase_create(<nama_test_case>)`. Test ditambahkan dengan `tcase_add_test(<variabel_test_case>, <nama_method_test>)`.
- **Suite**
  Suite merupakan tes yang akan dijalankan di main. Sebuah suite dapat dibuat dengan menggunakan `suite_create(<nama_suite>)`. Setelah itu, sebuah suite dapat ditambah test case baru dengan prosedur `suite_add_tcase(<variabel_suite>, <variabel_test_case>)`.

Menjalankan Test

Dengan Makefile

Cara menjalankan test yang paling mudah adalah dengan menggunakan Makefile yang telah disediakan oleh Check. Hal yang perlu diperhatikan adalah perubahan file *object*, yaitu:

- Ganti `<nama file>.o` dalam Makefile sesuai dengan nama file implementasi.
- Ganti `<nama_test_file>.o` dalam Makefile sesuai dengan nama file test.

Setelah itu lakukan salah satu dari kedua perintah ini:

- `$ make test`
  Ini akan berhenti pada testing dan memperlihatkan seberapa banyak testing yang benar.
- `$ make all`
  Setelah melakukan testing, maka akan membuat file html yang berisi laporan coverage yang berhasil di-*cover* oleh unit testing (jika lolos testing).

Tanpa Makefile

Ketikkan ini pada *terminal* atau *command line*:

- Membuat file *object* dari implementasi ADT
  `$ gcc -c -Wall -fprofile-arcs -ftest-coverage <nama_file>`
- Membuat file *object* dari file test (`check_<nama_file>`)
  `$ gcc -c -Wall -fprofile-arcs -ftest-coverage <nama_test_file>`
- Membuat executable file dari tes
  `$ gcc <nama_object_file> <nama_object_test_file> -lcheck -lm -lpthread -lrt -lsubunit -lgcov -coverage -o <executable_file>`
- Menjalankan executable file
  `$ ./<executable_file>`

Proses di atas merupakan proses yang dilakukan untuk mendapatkan hasil yang sesuai dengan `make test`.

Contoh Hasil Testing

Contoh akurasi testing

```
Running suite(s): Maksimum
50%: Checks: 6, Failures: 3, Errors: 0
tests/check_maksimum.c:6:F:Core:test_maksimum_1:0: Assertion 'maksimum3(1, 2,
3) == 2' failed: maksimum3(1, 2, 3) == 3, 2 == 2
tests/check_maksimum.c:10:F:Core:test_maksimum_2:0: Assertion 'maksimum3(1, 3,
 2) == 2' failed: maksimum3(1, 3, 2) == 3, 2 == 2
tests/check_maksimum.c:14:F:Core:test_maksimum_3:0: Assertion 'maksimum3(3, 2,
 1) == 2' failed: maksimum3(3, 2, 1) == 3, 2 == 2
```

Contoh hasil coverage

GCC Code Coverage Report

./		Exec	Total	
2020-08-26 22:59:25	Lines:	46	47	
low: < 75.0 % medium: >= 75.0 % high: >= 90.0 %	Branches:	11	18	

File	Lines			Branches	
src/maksimum.c		87.5 %	7 / 8	83.3 %	5 / 6
tests/check_maksimum.c		100.0 %	39 / 39	50.0 %	6 / 12