Secure and Dependable Systems
Jacobs University Bremen
Dr. Jürgen Schönwälder

Module: CO-566
Date: 2021-02-26
Due: 2021-03-05

SADS 2021 Problem Sheet #3

Problem 3.1: *reliability* (3+1+1 = 5 points)

Consider a computing system with the following component structure:

```
           .------.        .------.
     .-->--|  a   |-->-----|  b   |--.
     |     '------'        '------'  |     .------.
     |                               +-->--|  e   |-->--.
     |     .------.        .------.  |     '------'     |
-->--+-->--|  c   |-->--+--|  d   |--'                  |
     |     '------'    /   '------'                     +-->--
     |                /                                 |
     |     .------.  /     .------.                     |
     '-->--|  f   |-+--->--|  g   |-->------------------'
           '------'        '------'
```

a) Assuming that failures are independent, derive a formula for the reliability of the system. You can write down the formula in either plain math or as a function definition in Haskell.

b) Assuming all components have the same reliabily, generate a plot showing the system reliability as a function of the component reliability.

c) Assume all components have a reliability of 0.8. You have the financial resources to replace one component with a component that has a reliability of 0.95. Which component do you replace to maximize the system reliability? What is the system reliability you can achieve?

Problem 3.2: *test coverages* (1+1+1+1+1 = 5 points)

Below is C program to calculate the gcd of two integers.

```c
#include <stdio.h>
#include <stdlib.h>

int
gcd_euklid(int a, int b)
{
    while ((a > 0) && (b > 0)) {
        if (a > b) {
            a -= b;
        } else {
            b -= a;
        }
    }
    return a + b;
}

int
main(int argc, char *argv[])
{
    if (argc != 3) {
        fprintf(stderr, "%s: wrong number of arguments\n", argv[0]);
        return EXIT_FAILURE;
    }
    printf("%d\n", gcd_euklid(atoi(argv[1]), atoi(argv[2])));
    return EXIT_SUCCESS;
}
```

Lets assume the program has been compiled into the executable file `gcd`. Your task is to write down a *minimal* number of calls of the program (shell commands) that achieve different code coverages.

a) Which calls are necessary to achieve function coverage?

b) Which calls are necessary to achieve statement coverage?

c) Which calls are necessary to achieve branch coverage?

d) Which calls are necessary to achieve path coverage?

e) Which calls are necessary to achieve condition coverage?