Assignment 3

Batuhan Faik Derinbay
150180705
derinbay18@itu.edu.tr

Psuedocodes of Increaser and Decreaser Processes

Psuedocode of Increaser Processes

```
if (process == increaser_10){   // Or increaser_15
    sem_wait(increment_turn);
    sem_wait(money_box);
    money += 10;     // Or 15 depending on the process
    increaser_count ++; // Semaphore can be used instead of integer
    if (increaser_count == Ni){
        increaser_turn ++;
        increaser_count = 0;
        if (money >= N && pass == 0){
            pass = 1;   // Money is now above threshold and ↩
                decrementer processes can be signalled
        }
        // Check if increaser processes completed their turn
        if (increaser_turn % ti == 0 && pass){
            sem_post(decrement_turn, Nd);
        } else {
            sem_post(increment_turn, Ni);
        }
        sem_post(increaser_wait, Ni);   // Signal all increaser ↩
            processes to continue
    }
    sem_post(money_box);
    sem_wait(increaser_wait);   // Wait for continue signal
}
```

Psuedocode of Increaser Processes

```
if (process == decreaser_odd){   // Or decreaser_even
    sem_wait(decrement_turn);
    sem_wait(money_box);
    // Decrement only if money is odd (or even)
    if (money % 2 == 1){ // Or mod 2 = 0
        temp = money - fibonacci_number;
        if (temp < fibonacci_number){   // Not enough money, exit program
            sem_post(master);
        } else {
            money = temp;
        decreaser_count ++; // Semaphore can be used instead of integer
        if (decreaser_count == Nd){
            decreaser_turn ++;
            decreaser_count = 0;
            // Check if decreaser processes completed their turn
            if (decreaser_turn % td == 0){
                sem_post(increment_turn, Ni);
            } else {
                sem_post(decrement_turn, Nd);
            }
            sem_post(decreaser_waitm Nd);   // Signal all decreaser processes to continue
        }
        }
        // Get the next fibonacci number
        fibonacci_number = next_fibonacci(fibonacci_number);
    } else {    // If the process can't decrease money because their type (odd/even) doesn't match
        // Increase decreaser count and send all the necessary signals so decreasers can continue
        decreaser_count ++;
        if (decreaser_count == Nd){
        decreaser_turn ++;
        decreaser_count = 0;
        // Check if decreaser processes completed their turn
        if (decreaser_turn % td == 0){
            sem_post(increment_turn, Ni);
        } else {
            sem_post(decrement_turn, Nd);
        }
        sem_post(decreaser_waitm Nd);   // Signal all decreaser processes to continue
        }
    }
    sem_post(money_box);
    sem_wait(decreaser_wait);   // Wait for continue signal
}
```

Instructions for Compiling and Running

- Extract all the source code files (hw3.c is the only source code file) in a folder.

- Change your working directory to the folder that you've extracted the source code files.
- Write the following commands in order in the terminal. These commands will first compile the source code, then they will run the program with your inputs. Please note that variables N, Ni, Nd, ti and td are explained in the homework description. You need to change them to your preffered values.
  - `gcc hw3.c -o hw3 -std=c99`
  - `./hw3 N ni nd ti td`
- An example to run the program
  - `./hw3 150 4 2 2 4`