

Sprawozdanie

Zadanie 2

Zadanie brzmi «Stworzenie programu tworzącego dwa wątki – jeden inkrementuje zmienną licznik w strukturze danych do 100000, a drugi ją dekrementuje tyle samo razy. Jaką wartość ma na końcu ta zmienna? – rozwiąż problem by dawała wartość 0»

Kod źródłowy rozwiązania:

```
#include <stdio.h>
#include <pthread.h>

struct Counter {
    int licznik;
    pthread_mutex_t lock;
};

struct Counter c;

void *incrementCounter() {
    pthread_mutex_lock(&c.lock);

    for (int i = 0; i < 1e5; ++i) {
        ++c.licznik;
    }

    pthread_mutex_unlock(&c.lock);

    return NULL;
}

void *decrementCounter() {
    pthread_mutex_lock(&c.lock);

    for (int i = 0; i < 1e5; ++i) {
        --c.licznik;
    }

    pthread_mutex_unlock(&c.lock);

    return NULL;
}
```

```

}

int main() {
    pthread_t tid[2];

    pthread_mutex_init(&c.lock, NULL);

    pthread_create(&tid[0], NULL, incrementCounter, NULL);
    pthread_create(&tid[1], NULL, decrementCounter, NULL);

    for (short i = 0; i < 2; ++i) {
        pthread_join(tid[i], NULL);
    }

    pthread_mutex_destroy(&c.lock);

    printf("Final result is %d\n", c.licznik);
    return 0;
}

```

Wydruk programu:

```
Final result is 0
```

Zadanie 3

Zadanie brzmi «Zaimplementować symulację pubu przy powyższych założeniach.

Parametrami symulacji są: ilość klientów, ilość kufli oraz maksymalny czas picia piwa.»

Założenia są:

- W pubie jest n kufli 1 L.
- Klienci są reprezentowani przez wątki.
- Kufli jest mniej niż klientów.
- Każdy klient pragnie wypić 2 litry piwa.
- Klient opróżnia kufel losową ilość czasu
- Po wypiciu 2 litrów klient opuszcza pub
- Pub otwarty jest do ostatniego klienta (ale nie wpuszcza nowych).

Kod źródłowy rozwiązania:

```

#include <stdlib.h>
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <time.h>

#define N 2 // Number of mugs
#define K 5 // Number of clients
#define GULP 5 // Maximum time needed to drink

struct Pub {
    pthread_mutex_t mugs[N];
};

struct ClientAttrs {
    struct Pub *pub;
    short id;
};

void *booze(void *attrs){
    struct ClientAttrs *clientAttrs = attrs;
    struct Pub *pub = clientAttrs->pub;
    short *clientId = &clientAttrs->id;

    for(int i = 0; i < 2; ++i) {
        short t = (rand() % (GULP + 1));
        short mugId = -1;

        // Find and take free mug
        while(mugId == -1) {
            for(short j = 0; j < N; ++j) {
                if(pthread_mutex_trylock(&pub->mugs[j]) == 0) {
                    mugId = j;
                    break;
                }

                if (j == N - 1) {
                    j = -1;
                }
            }
        }

        printf("Klient nr %d zamawia piwo w kuflu nr %d\n",
*clientId, mugId);
        fflush(stdout);
    }
}

```

```

        sleep(t); // Drink

        printf("Klient nr %d oddaje kufel nr %d po czasie %d\n",
*clientId, mugId, t);
        fflush(stdout);

        // Return mug
        pthread_mutex_unlock(&pub->mugs[mugId]);
    }

    return(NULL);
}

int main() {
    srand(time(0));
    struct Pub pub;
    pthread_t client[K];
    struct ClientAttrs clientAttrs[K];
    for (short i = 0; i < N; ++i) {
        pthread_mutex_init(&pub.mugs[i], NULL);
    }

    // Spawn clients
    for(short i = 0; i < K; ++i) {
        clientAttrs[i].id = i;
        clientAttrs[i].pub = &pub;

        pthread_create(&client[i], NULL, booze, &clientAttrs[i]);
    }

    // Wait for clients to get drunk
    for(short i = 0; i < K; ++i) {
        pthread_join(client[i], NULL);
    }

    for (short i = 0; i < N; ++i) {
        pthread_mutex_destroy(&pub.mugs[i]);
    }

    return 0;
}

```

Wydruk programu dla parametrów (ilość klientów = 5, ilość kufli = 2, maksymalny czas picia piwa = 5) :

```
Klient nr 0 zamawia piwo w kuflu nr 0
Klient nr 1 zamawia piwo w kuflu nr 1
Klient nr 1 oddaje kufel nr 1 po czasie 0
Klient nr 3 zamawia piwo w kuflu nr 1
Klient nr 3 oddaje kufel nr 1 po czasie 1
Klient nr 1 zamawia piwo w kuflu nr 1
Klient nr 0 oddaje kufel nr 0 po czasie 2
Klient nr 2 zamawia piwo w kuflu nr 0
Klient nr 2 oddaje kufel nr 0 po czasie 2
Klient nr 0 zamawia piwo w kuflu nr 0
Klient nr 1 oddaje kufel nr 1 po czasie 3
Klient nr 3 zamawia piwo w kuflu nr 1
Klient nr 3 oddaje kufel nr 1 po czasie 0
Klient nr 2 zamawia piwo w kuflu nr 1
Klient nr 0 oddaje kufel nr 0 po czasie 1
Klient nr 4 zamawia piwo w kuflu nr 0
Klient nr 4 oddaje kufel nr 0 po czasie 0
Klient nr 4 zamawia piwo w kuflu nr 0
Klient nr 2 oddaje kufel nr 1 po czasie 2
Klient nr 4 oddaje kufel nr 0 po czasie 5
```

Wydruk programu dla parametrów (ilość klientów = 7, ilość kufli = 5, maksymalny czas picia piwa = 7) :

```
Klient nr 0 zamawia piwo w kuflu nr 0
Klient nr 1 zamawia piwo w kuflu nr 1
Klient nr 2 zamawia piwo w kuflu nr 2
Klient nr 3 zamawia piwo w kuflu nr 3
Klient nr 4 zamawia piwo w kuflu nr 4
Klient nr 0 oddaje kufel nr 0 po czasie 0
Klient nr 6 zamawia piwo w kuflu nr 0
Klient nr 3 oddaje kufel nr 3 po czasie 0
Klient nr 5 zamawia piwo w kuflu nr 3
Klient nr 5 oddaje kufel nr 3 po czasie 0
Klient nr 3 zamawia piwo w kuflu nr 3
Klient nr 2 oddaje kufel nr 2 po czasie 1
Klient nr 5 zamawia piwo w kuflu nr 2
Klient nr 6 oddaje kufel nr 0 po czasie 2
Klient nr 2 zamawia piwo w kuflu nr 0
Klient nr 4 oddaje kufel nr 4 po czasie 3
```

```
Klient nr 6 zamawia piwo w kuflu nr 4
Klient nr 1 oddaje kufel nr 1 po czasie 4
Klient nr 0 zamawia piwo w kuflu nr 1
Klient nr 5 oddaje kufel nr 2 po czasie 4
Klient nr 4 zamawia piwo w kuflu nr 2
Klient nr 0 oddaje kufel nr 1 po czasie 1
Klient nr 1 zamawia piwo w kuflu nr 1
Klient nr 3 oddaje kufel nr 3 po czasie 7
Klient nr 2 oddaje kufel nr 0 po czasie 6
Klient nr 6 oddaje kufel nr 4 po czasie 7
Klient nr 1 oddaje kufel nr 1 po czasie 6
Klient nr 4 oddaje kufel nr 2 po czasie 7
```

Wydruk programu dla parametrów (ilość klientów = 4, ilość kufli = 3, maksymalny czas picia piwa = 2) :

```
Klient nr 0 zamawia piwo w kuflu nr 0
Klient nr 3 zamawia piwo w kuflu nr 1
Klient nr 2 zamawia piwo w kuflu nr 2
Klient nr 0 oddaje kufel nr 0 po czasie 1
Klient nr 1 zamawia piwo w kuflu nr 0
Klient nr 3 oddaje kufel nr 1 po czasie 1
Klient nr 0 zamawia piwo w kuflu nr 1
Klient nr 2 oddaje kufel nr 2 po czasie 2
Klient nr 3 zamawia piwo w kuflu nr 2
Klient nr 1 oddaje kufel nr 0 po czasie 1
Klient nr 2 zamawia piwo w kuflu nr 0
Klient nr 0 oddaje kufel nr 1 po czasie 1
Klient nr 1 zamawia piwo w kuflu nr 1
Klient nr 2 oddaje kufel nr 0 po czasie 1
Klient nr 3 oddaje kufel nr 2 po czasie 2
Klient nr 1 oddaje kufel nr 1 po czasie 2
```