Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

Операційні системи

Лабораторна №8

Виконав:
Студент групи ФБ-82
Козачок Вячеслав
Перевірив:
Кіреєнко О.В.

Київ - 2020

0.1 Code

```
#include <iostream>
#include <pthread.h>
#include <queue>
#include <semaphore.h>
#include <time.h>
#include <unistd.h>

struct thread_info
{
    int thread_number;
    char * name;
    pthread_t * thread;
};

std::queue<thread_info*> on_board;
std::queue<thread_info*> to_cancel;

sem_t busy_places, boarding, disembarkation;

int paratroopers_on_board = 0;
int marines_on_board = 0;
int max_places;
int handled_soldiers = 0;

void trip()
{
    printf("\n");
    for(int i = 0; i < 3; i++)
    {
        printf("%16s\n", "Ferry in trip!");
        sleep(1);
    }
    printf("\n");

    // Allow troops leave the ferry
    sem_post(&disembarkation);
}

void get_on_board(thread_info * info)
{
    sem_wait(&boarding);
    if(info->name == "Marine")
    {
        if((marines_on_board == max_places/2 && paratroopers_on_board != 0) || (
    paratroopers_on_board > max_places/2))
        {
            printf("%12s %d canceled ", info->name, info->thread_number);
            to_cancel.push(info);
        }
        else
        {
            printf("%12s %d boarded  ", info->name, info->thread_number);
            on_board.push(info);
            marines_on_board++;
        }
    }
    else if (info->name == "Paratrooper")
    {
        if((paratroopers_on_board == max_places/2 && marines_on_board != 0) || (
    marines_on_board > max_places/2))
        {
            printf("%12s %d canceled ", info->name, info->thread_number);
            to_cancel.push(info);
        }
        else
        {
            printf("%12s %d boarded  ", info->name, info->thread_number);
            on_board.push(info);
            paratroopers_on_board++;
        }
    }
```

```
70
71      printf("< Current Time: %d \n", time(0));
72      handled_soldiers++;
73      sleep(1);
74      sem_post(&boarding);
75  }
76
77  void get_out_of_board(thread_info * info)
78  {
79      sem_wait(&disembarkation);
80      printf("%12s %d disembarked. < Current time: %d\n", info->name, info->thread_number,
        time(0));
81      sleep(1);
82      sem_post(&disembarkation);
83  }
84
85  void clear_ferry()
86  {
87      while(on_board.size() != 0)
88      {
89          // printf("Waiting clear\n");
90          pthread_join(*(on_board.front()->thread), NULL);
91          on_board.pop();
92      }
93  }
94
95  void cancel_threads()
96  {
97      while(to_cancel.size() != 0)
98      {
99          pthread_cancel(*(to_cancel.front()->thread));
100         to_cancel.pop();
101     }
102 }
103
104
105 void * thread_work(void * param)
106 {
107     thread_info * info = (thread_info*)param;
108     sleep(rand() % 3 + 1);
109
110     get_on_board(info);
111     get_out_of_board(info);
112     pthread_exit(NULL);
113 }
114
115 int main(int argv, char * argc[])
116 {
117     std::cout << "Enter N: ";
118     int n;
119     std::cin >> n;
120     if(std::cin.fail())
121     {
122         std::cout << "Exiting ...\n";
123         return 1;
124     }
125
126     srand(time(0));
127     max_places = n * 2;
128     printf("\n%16s\n", "INFORMATION");
129     printf("%16s: %d\n", "Available places", max_places);
130     printf("%16s: %d\n", "Paratroopers", max_places);
131     printf("%16s: %d\n", "Marins", max_places);
132     printf("%20s\n", "------------");
133
134     // Semaphores
135     sem_init(&boarding, 0, 1);
136     sem_init(&disembarkation, 0, 0);
137
138     // Soldiers
139     pthread_t * marine = new pthread_t[max_places];
140     pthread_t * paratrooper = new pthread_t[max_places];
141
```

```
    int counter = 0;
    thread_info * info;
    for(int i = 0; i < max_places; i++)
    {
        // Creating marine
        info = new thread_info;
        info->thread_number = counter;
        info->thread = &marine[i];
        info->name = "Marine";
        pthread_create(&marine[i], NULL, thread_work, (void*)info);

        // Creating paratrooper
        info = new thread_info;
        info->thread_number = counter;
        info->thread = &paratrooper[i];
        info->name = "Paratrooper";
        pthread_create(&paratrooper[i], NULL, thread_work, ((void*)info));
        counter++;
    }

    // wait till all soldiers is canceled or boarded
    while(handled_soldiers != max_places*2)
    {
        sleep(1);
        continue;
    }

    // remove soldiers that could not board ferry
    printf("\nCleaning all threads troops that are not in ferry\n");
    cancel_threads();

    trip();

    // clear ferry on another side
    printf("Eject all troops on coast\n\n");
    clear_ferry();

    return 0;
}
```

Output

N = 2

```
Enter N: 2

     INFORMATION
Available places: 4
    Paratroopers: 4
          Marins: 4
        ------------
 Paratrooper 0 boarded  < Current Time: 1588159550
 Paratrooper 2 boarded  < Current Time: 1588159551
      Marine 0 boarded  < Current Time: 1588159552
 Paratrooper 1 canceled < Current Time: 1588159553
      Marine 2 boarded  < Current Time: 1588159554
      Marine 3 canceled < Current Time: 1588159555
 Paratrooper 3 canceled < Current Time: 1588159556
      Marine 1 canceled < Current Time: 1588159557

Cleaning all threads troops that are not in ferry

  Ferry in trip!
  Ferry in trip!
  Ferry in trip!

Eject all troops on coast

 Paratrooper 0 disembarked. < Current time: 1588159561
 Paratrooper 2 disembarked. < Current time: 1588159562
      Marine 0 disembarked. < Current time: 1588159563
      Marine 2 disembarked. < Current time: 1588159564
```