OS2 Lab3

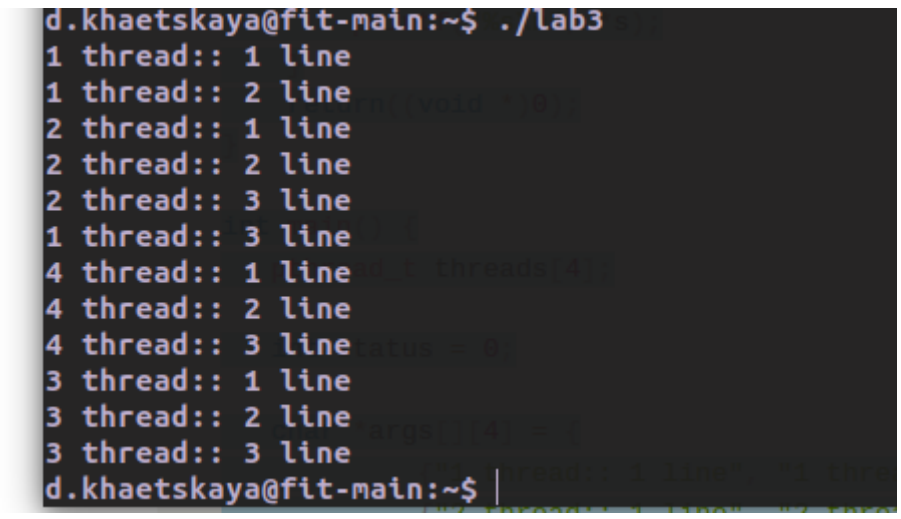
checkbox	☑
due date	@September 17, 2021
class	OS
Status	approved

Task

Напишите программу, которая создает четыре нити, исполняющие одну и ту же функцию. Эта функция должна распечатать последовательность текстовых строк, переданных как параметр. Каждая из созданных нитей должна распечатать различные последовательности строк.

Notes

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

void *printString(void *args) {
    for (char **s = (char **)args; *s != NULL; ++s) {
        printf("%s\n", *s);
    }
    return((void *)0);
}

int main() {
   pthread_t threads[4];

   int status = 0;

   char *args[][4] = {
            {"1 thread:: 1 line", "1 thread:: 2 line", "1 threa
d:: 3 line", NULL},
            {"2 thread:: 1 line", "2 thread:: 2 line", "2 threa
d:: 3 line", NULL},
            {"3 thread:: 1 line", "3 thread:: 2 line", "3 threa
d:: 3 line", NULL},
            {"4 thread:: 1 line", "4 thread:: 2 line", "4 threa
d:: 3 line", NULL}
    };


  for (int i = 0; i < 4; i++){
      status = pthread_create(&threads[i], NULL, printString,
 (void*)args[i]);

      if (status != 0){
          perror("failed to create thread");
          exit(-1);
      }
  }

   for (int i = 0; i < 4; i++){
       status = pthread_join(threads[i], NULL);

       if (status != 0){
           perror("failed to join thread");
           exit(-1);
       }
   }

   return 0;
}
```



По умолчанию код завершения потока сохраняется, пока для этого потока не будет вызвана функция `pthread_join`. Основная память потока может быть немедленно освобождена по его завершении, если поток был обособлен. Когда поток обособлен, функция `pthread_join` не может использоваться для получения его кода завершения, потому что в этом случае ее поведение не определено. Обособить поток можно с помощью функции `pthread_detach`.

При передаче структур данных в качестве параметра нужно проявлять осторожность.

1. **Не следует передавать структуры данных, размещенные в стеке родительской нити,** то есть переменные с классом памяти auto и блоки памяти, размещенные при помощи alloca(3C). Если нить вернёт управление из текущей функции или завершится до того, как созданная нить начнет работать с блоком параметров, получится, что в качестве параметра была передана висячая ссылка.
2. При передаче структур данных, размещенных **при помощи malloc(3C)**, необходимо решить вопрос о том, **кто будет освобождать эту структуру**. Если структура не будет освобождена при помощи free(3C), это приведет к утечке памяти. Существует несколько решений этого вопроса:

   Первое решение: родитель размещает структуру, а созданная нить ее освобождает. Но обычно это считается дурным тоном при программировании на C/C++. *Хорошим тоном* считается, чтобы вызываемая функция не знала, каким образом выделена память под переданные ей параметры.

   Второе решение: родитель размещает структуру, дожидается завершения потомка при помощи вызова pthread_join(3C) и освобождает структуру. *В рамках этого подхода можно передавать структуры данных, созданные при помощи malloc(3C) или размещенные в стеке родителя.* Нужно помнить, что если родитель будет принудительно завершен при помощи **pthread_cancel(3C)**, он может **не дождаться завершения своих потомков**, а это приведет либо к утечке памяти, либо к висячим ссылкам.

В программах с небольшим количеством нитей часто передают в качестве параметра указатели на статические переменные, но в больших программах с большим числом нитей это *неприемлемо*.