****

**Operating System**

**Lab # 14**

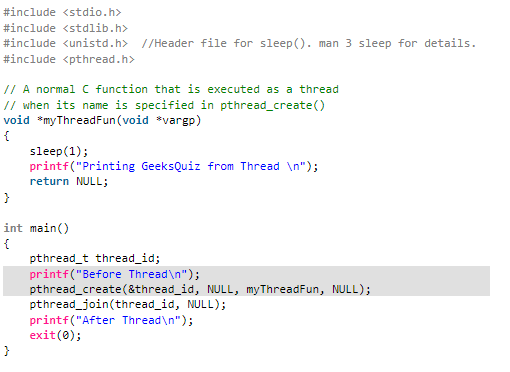
**Name: Zainab Bibi**

**Sap Id: 46462**

**Batch: BSCS 5thsemester**

**Submitted to: Mam Kausar**

**Question: 1**



**Solution:**

**Libraries**

* **stdio.h**: Handles input and output operations (e.g., printf).
* **stdlib.h**: Provides utility functions like exit().
* **unistd.h**: Supports system calls such as sleep().
* **pthread.h**: Manages threads (e.g., pthread\_create, pthread\_join).

**myThreadFun Function**

* Serves as the thread's task function.
* Pauses for 1 second (sleep(1)).
* Displays "Printing GeeksQuiz from Thread".
* Returns NULL as it doesn’t provide any output.

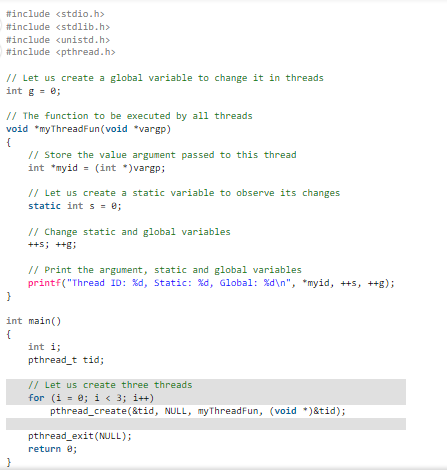
**main Function**

* Initializes a thread with pthread\_create.
* Waits for the thread to finish using pthread\_join.
* Prints text before starting the thread and after it completes.

**Output Sequence**

* The displayed result will be:
  + Before Thread
  + Printing GeeksQuiz from Thread
  + After Thread

**Question: 2**



**Solution:**

**Header Files**

* stdio.h: For input/output functions (e.g., printf).
* stdlib.h: For utility functions (e.g., exit()).
* unistd.h: For system functions like sleep().
* pthread.h: To work with threads (pthread\_create, pthread\_exit).

**Global and Static Variables**

* **Global Variable (g)**: Shared across all threads, initialized to 0.
* **Static Variable (s)**: Retains its value across function calls, initialized to 0.

**myThreadFun Function**

* Accepts a void\* argument, cast to an integer pointer (myid).
* Increments the static variable s and the global variable g.
* Prints the thread ID, the value of s, and g for each thread.
* Shows how global and static variables behave in a multithreaded environment.

**main Function**

1. Declares i (loop counter) and tid (thread ID).
2. Uses a for loop to create **3 threads** using pthread\_create.
   * Each thread executes myThreadFun with tid as an argument.
3. Ends the program with pthread\_exit(NULL) to wait for all threads to finish.

**Step-by-Step Explanation:**

**Thread Creation**

* Three threads are launched using a loop.
* The loop passes the current index (i) as an argument to each thread.

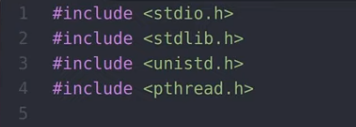
**Behavior of Shared Variables**

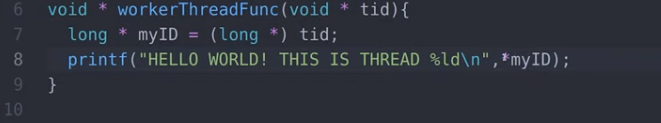
* **Static variable (s)**: Retains its value across multiple calls within the same thread, as it is local to the function but shared within its scope.
* **Global variable (g)**: Accessible and modified by all threads, increasing each time a thread executes.

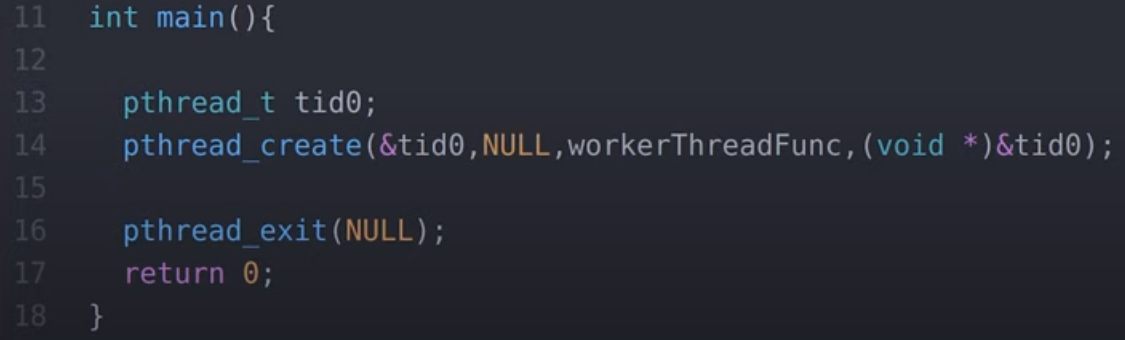
**Output Details**

* Each thread outputs its unique ID, the value of s, and the updated value of the global variable g.
* The sequence of output may vary depending on how the operating system schedules the threads.

**Question: 3**







**Solution:**

**Thread Execution Overview**

* A new thread is created to perform tasks concurrently with the main program.
* This thread runs a function (workerThreadFunc()) that displays a message along with its unique thread ID.

**Important Code Highlights**

* pthread\_create(&tid0, NULL, workerThreadFunc, (void\*)&tid0);: Initiates the thread and passes its ID as an argument.
* pthread\_exit(NULL);: Ensures the main program remains active until the thread completes its execution.

**Functionality of the Worker Thread**

* The thread executes a function that outputs the message:  
  "HELLO WORLD! THIS IS THREAD X", where X is the unique thread ID assigned by the system.

**Purpose of pthread\_exit()**

* Prevents the main program from terminating before the thread has completed, ensuring proper execution of the thread's task.

**Expected Output**

* The output will be:  
  "HELLO WORLD! THIS IS THREAD <Thread\_ID>"  
  For example: "HELLO WORLD! THIS IS THREAD 139885472502016".

**Question: 4**

Define posix thread and its working in your own words?

**Solution:**

**POSIX Threads (Pthreads)**  
POSIX Threads are a standard way to create and manage multiple threads in a program. Threads run tasks at the same time, sharing the same memory space.

**How They Work:**

* A thread is like a smaller part of a program that runs on its own.
* Pthreads allow you to create threads, pass data to them, and make them do tasks.
* The program can wait for threads to finish using special functions.

It’s useful for doing many tasks at once, like downloading files while showing progress.