Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

**12**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| 1 | **Create five numbers of thread after creation release them, make a function that acquired all five threads one by one, and put a releasing message after every mutex** |
| 2 | **Using mutex print odd and even’s cube within the range of 20.** |

Submitted On:

**Task # 01: Create five numbers of thread after creation release them, make a function that acquired all five threads one by one, and put a releasing message after every mutex**

**Solution:**

#include<iostream>

#include<thread>

#include<mutex>

std::mutex m;

void ThreadFunc(int threadid)

{

m.lock();

std::cout << "Thread " << threadid << " acquired the mutex" << std::endl;

m.unlock();

std::cout << "Thread " << threadid << " released the mutex" << std::endl;

}

int main()

{

std::thread threads[5];

//Create 5 threads

for (int i = 0; i < 5; i++)

{

threads[i] = std::thread(ThreadFunc, i + 1);

}

//Join each thread

for (int i = 0; i < 5; i++)

{

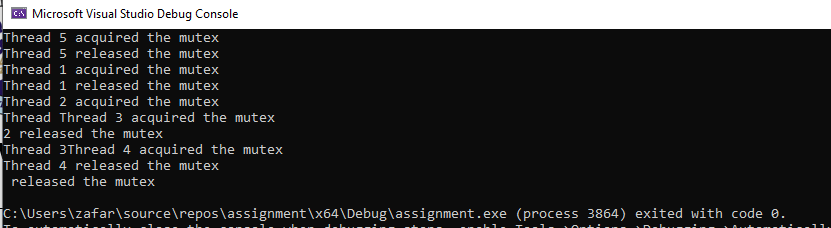
threads[i].join();

}

return 0;

}

Output:



**Task # 02: Using mutex print odd and even’s cube within the range of 20**

**Solution:**

#include <iostream>

#include <thread>

#include <mutex>

std::mutex mutex;

void print\_even\_cubes(int n)

{

for (int i = 0; i <= n; i += 2)

{

mutex.lock();

std::cout << "Cube of " << i << " is: " << i \* i \* i << std::endl;

mutex.unlock();

}

}

void print\_odd\_cubes(int n)

{

for (int i = 1; i <= n; i += 2)

{

mutex.lock();

std::cout << "Cube of " << i << " is: " << i \* i \* i << std::endl;

mutex.unlock();

}

}

int main()

{

int n = 20;

std::thread t1(print\_even\_cubes, n);

std::thread t2(print\_odd\_cubes, n);

t1.join();

t2.join();

return 0;

}

Output:

