CEC14 OPERATING SYSTEM

TUTORIAL

2018UCO1550

NIPUNIKA

TABLE OF CONTENTS

Program 1: Process Creation and Termination

CODE:

```
#include <iostream>
#include <sys/wait.h>
#include <unistd.h>

using namespace std;

int main()
{
   pid_t id1 = fork();
   pid_t id2 = fork();
    if (id1 > 0 && id2 > 0)
   {
       wait(NULL);
       wait(NULL);
       cout << "Parent Terminated" << endl;
   }
    else if (id1 == 0 && id2 > 0)
   {
       sleep(2);
       wait(NULL);
       cout << "1st child Terminated" << endl;
   }
    else if (id1 > 0 && id2 == 0)
   {
       sleep(1);
       cout << "2nd Child Terminated" << endl;
   }
    else
   {
       cout << "Grand Child Terminated" << endl;
   }
    return 0;
```

```
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ creation_termination.c
pp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Grand Child Terminated
2nd Child Terminated
1st child Terminated
Parent Terminated
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

Program 2: Producer Consumer Problem using bounded and unbounded buffer

CODE:

```
#include<stdio.h>
#include<stdlib.h>
int mutex=1,full=0,empty=3,x=0;
int wait(int s)
{
   return (--s);
}
int signal(int s)
{
   return(++s);
}
void producer()
{
   mutex=wait(mutex);
   full=signal(full);
   empty=wait(empty);
   x++;
   printf("\nProducer produces the item %d",x);
   mutex=signal(mutex);
}
void consumer()
{
   mutex=wait(mutex);
   full=wait(full);
   empty=signal(empty);
   printf("\nConsumer consumes item %d",x);
   x--;
   mutex=signal(mutex);
}
```

```c
int main()
{
   int n;
   void producer();
   void consumer();
   int wait(int);
   int signal(int);
   printf("\n1.Producer\n2.Consumer\n3.Exit");
   while(1)
   {
       printf("\nEnter your choice:");
       scanf("%d",&n);
       switch(n)
       {
           case 1: if((mutex==1)&&(empty!=0))
                       producer();
                   else
                       printf("Buffer is full!!");
                   break;
           case 2: if((mutex==1)&&(full!=0))
                       consumer();
                   else
                       printf("Buffer is empty!!");
                   break;
           case 3:
                   exit(0);
                   break;
       }
   }

   return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ prod_cons.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out

1.Producer
2.Consumer
3.Exit
Enter your choice:1

Producer produces the item 1
Enter your choice:2

Consumer consumes item 1
Enter your choice:1

Producer produces the item 1
Enter your choice:22

Enter your choice:2

Consumer consumes item 1
Enter your choice:2
Buffer is empty!!
Enter your choice:21
```

Program 3: Interprocess Communication

CODE:

```
// For the writer process
#include <stdio.h>
#include <iostream>
#include <sys/ipc.h>
#include <sys/msg.h>

struct mesg_buffer
{
   long mesg_type;
   char mesg_text[100];
} message;
 int main()
{
   key_t key;
   int msgid;
    key = ftok("progfile", 65);
    msgid = msgget(key, 0666 | IPC_CREAT);
   message.mesg_type = 1;
    printf("Write Data : ");
   std::cin >> message.mesg_text;
    msgsnd(msgid, &message, sizeof(message), 0);
    printf("Data send is : %s \n", message.mesg_text);
    return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Write Data : 123
Data send is : 123
```

Program 4: CPU Scheduling Algorithms

First Come First Serve(FCFS)

CODE:

```
#include<iostream>

using namespace std;
 void findWaitingTime(int processes[], int n, int bt[], int wt[])
{
   wt[0] = 0;
    for (int  i = 1; i < n ; i++ )
       wt[i] =  bt[i-1] + wt[i-1] ;
}
 void findTurnAroundTime( int processes[], int n, int bt[], int wt[], int
tat[])
{
   for (int  i = 0; i < n ; i++)
       tat[i] = bt[i] + wt[i];
}
 void findavgTime( int processes[], int n, int bt[])
{
   int wt[n], tat[n], total_wt = 0, total_tat = 0;
    findWaitingTime(processes, n, bt, wt);
    findTurnAroundTime(processes, n, bt, wt, tat);
    cout << "Processes  "<< " Burst time  " << " Waiting time  " << " Turn
around time\n";
    for (int  i=0; i<n; i++)
   {
       total_wt = total_wt + wt[i];
       total_tat = total_tat + tat[i];
       cout << "   " << i+1 << "\t\t" << bt[i] <<"\t    " << wt[i] <<"\t\t
" << tat[i] <<endl;
   }
    cout << "Average waiting time = " << (float)total_wt / (float)n;
```

```
   cout << "\nAverage turn around time = " << (float)total_tat / (float)n;
}
 int main()
{
   int processes[] = { 1, 2, 3};
   int n = sizeof processes / sizeof processes[0];
    int  burst_time[] = {10, 5, 8};
    findavgTime(processes, n,  burst_time);
   return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ CPU_FCFS.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a,out
bash: ./a,out: No such file or directory
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Processes   Burst time   Waiting time   Turn around time
   1            10           0              10
   2            5            10             15
   3            8            15             23
Average waiting time = 8.33333
Average turn around time = 16nipunika@nipunika-Nitro-AN515-52:~/Downloads/O
```

Shortest Job First

CODE:

```
#include<iostream>

using namespace std;

int mat[10][6];
 void swap(int *a, int *b)
{
   int temp = *a;
   *a = *b;
   *b = temp;
```

```
}
 void arrangeArrival(int num, int mat[][6])
{
   for(int i=0; i<num; i++)
   {
       for(int j=0; j<num-i-1; j++)
       {
           if(mat[j][1] > mat[j+1][1])
           {
               for(int k=0; k<5; k++)
               {
                   swap(mat[j][k], mat[j+1][k]);
               }
           }
       }
   }
}
 void completionTime(int num, int mat[][6])
{
   int temp, val;
   mat[0][3] = mat[0][1] + mat[0][2];
   mat[0][5] = mat[0][3] - mat[0][1];
   mat[0][4] = mat[0][5] - mat[0][2];

   for(int i=1; i<num; i++)
   {
       temp = mat[i-1][3];
       int low = mat[i][2];
       for(int j=i; j<num; j++)
       {
           if(temp >= mat[j][1] && low >= mat[j][2])
           {
               low = mat[j][2];
               val = j;
           }
       }
       mat[val][3] = temp + mat[val][2];
```

```
        mat[val][5] = mat[val][3] - mat[val][1];
        mat[val][4] = mat[val][5] - mat[val][2];
        for(int k=0; k<6; k++)
        {
            swap(mat[val][k], mat[i][k]);
        }
    }
}
 int main()
{
    int num, temp;

    cout<<"Enter number of Process: ";
    cin>>num;

    cout<<"...Enter the process ID...\n";
    for(int i=0; i<num; i++)
    {
        cout<<"...Process "<<i+1<<"...\n";
        cout<<"Enter Process Id: ";
        cin>>mat[i][0];
        cout<<"Enter Arrival Time: ";
        cin>>mat[i][1];
        cout<<"Enter Burst Time: ";
        cin>>mat[i][2];
    }

    cout<<"Before Arrange...\n";
    cout<<"Process ID\tArrival Time\tBurst Time\n";
    for(int i=0; i<num; i++)
    {
        cout<<mat[i][0]<<"\t\t"<<mat[i][1]<<"\t\t"<<mat[i][2]<<"\n";
    }

    arrangeArrival(num, mat);
    completionTime(num, mat);
    cout<<"Final Result...\n";
```

```
   cout<<"Process ID\tArrival Time\tBurst Time\tWaiting Time\tTurnaround
Time\n";
   for(int i=0; i<num; i++)
   {

cout<<mat[i][0]<<"\t\t"<<mat[i][1]<<"\t\t"<<mat[i][2]<<"\t\t"<<mat[i][4]<<
"\t\t"<<mat[i][5]<<"\n";
   }
}
```

OUTPUT:

```
Average turn around time = 16nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tut
s$ g++ CPU_SJF.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Enter number of Process: 2
...Enter the process ID...
...Process 1...
Enter Process Id: 23
Enter Arrival Time: 23
Enter Burst Time: 24
...Process 2...
Enter Process Id: 54
Enter Arrival Time: 34
Enter Burst Time: 54
Before Arrange...
Process ID      Arrival Time    Burst Time
23              23              24
54              34              54
Final Result...
Process ID      Arrival Time    Burst Time      Waiting Time    Turnaround Time
23              23              24              0               24
54              34              54              13              67
```

Priority Scheduling

CODE:

```
#include <iostream>
#include <algorithm>

using namespace std;
```

```
 struct Process
{
   int pid;
   int bt;
   int priority;
};
 bool comparison(Process a, Process b)
{
   return (a.priority > b.priority);
}
 void findWaitingTime(Process proc[], int n, int wt[])
{
   wt[0] = 0;
    for (int  i = 1; i < n ; i++ )
       wt[i] =  proc[i-1].bt + wt[i-1] ;
}
 void findTurnAroundTime( Process proc[], int n, int wt[], int tat[])
{
   for (int  i = 0; i < n ; i++)
       tat[i] = proc[i].bt + wt[i];
}
 void findavgTime(Process proc[], int n)
{
   int wt[n], tat[n], total_wt = 0, total_tat = 0;
    findWaitingTime(proc, n, wt);
    findTurnAroundTime(proc, n, wt, tat);
    cout << "\nProcesses  "<< " Burst time  " << " Waiting time  " << "
Turn around time\n";

   for (int  i=0; i<n; i++)
   {
       total_wt = total_wt + wt[i];
       total_tat = total_tat + tat[i];
       cout << "   " << proc[i].pid << "\t\t" << proc[i].bt << "\t    " <<
wt[i] << "\t\t  " << tat[i] <<endl;
   }
    cout << "\nAverage waiting time = " << (float)total_wt / (float)n;
```

```
   cout << "\nAverage turn around time = " << (float)total_tat / (float)n;
}
 void priorityScheduling(Process proc[], int n)
{
   sort(proc, proc + n, comparison);
    cout<< "Order in which processes gets executed \n";
   for (int  i = 0 ; i <  n; i++)
       cout << proc[i].pid <<" " ;
    findavgTime(proc, n);
}

int main()
{
   Process proc[] = {{1, 10, 2}, {2, 5, 0}, {3, 8, 1}};
   int n = sizeof proc / sizeof proc[0];
   priorityScheduling(proc, n);
   return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Order in which processes gets executed
1 3 2
Processes   Burst time   Waiting time   Turn around time
   1            10           0              10
   3            8            10             18
   2            5            18             23

Average waiting time = 9.33333
Average turn around time = 17nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tu
s$ _
```

Round Robin

CODE:

```
#include<iostream>
```

```
using namespace std;
 void findWaitingTime(int processes[], int n, int bt[], int wt[], int
quantum)
{
   int rem_bt[n];
   for (int i = 0 ; i < n ; i++)
       rem_bt[i] =  bt[i];
    int t = 0;

   while (1)
   {
       bool done = true;

       for (int i = 0 ; i < n; i++)
       {
           if (rem_bt[i] > 0)
           {
               done = false;
                if (rem_bt[i] > quantum)
               {
                   t += quantum;
                    rem_bt[i] -= quantum;
               }
                else
               {
                   t = t + rem_bt[i];
                    wt[i] = t - bt[i];
                    rem_bt[i] = 0;
               }
           }
       }
        if (done == true)
         break;
   }
}
 void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int
tat[])
```

```
{
   for (int i = 0; i < n ; i++)
       tat[i] = bt[i] + wt[i];
}
 void findavgTime(int processes[], int n, int bt[], int quantum)
{
   int wt[n], tat[n], total_wt = 0, total_tat = 0;
    findWaitingTime(processes, n, bt, wt, quantum);
    findTurnAroundTime(processes, n, bt, wt, tat);
    cout << "Processes "<< " Burst time " << " Waiting time " << " Turn
around time\n";

   for (int i=0; i<n; i++)
   {
       total_wt = total_wt + wt[i];
       total_tat = total_tat + tat[i];
       cout << " " << i+1 << "\t\t" << bt[i] <<"\t " << wt[i] <<"\t\t " <<
tat[i] <<endl;
   }
    cout << "Average waiting time = " << (float)total_wt / (float)n;
   cout << "\nAverage turn around time = " << (float)total_tat / (float)n;
}
 int main()
{
   int processes[] = { 1, 2, 3};
   int n = sizeof processes / sizeof processes[0];
    int burst_time[] = {10, 5, 8};
    int quantum = 2;
   findavgTime(processes, n, burst_time, quantum);
   return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Order in which processes gets executed
1 3 2
Processes   Burst time   Waiting time   Turn around time
   1            10           0              10
   3            8            10             18
   2            5            18             23

Average waiting time = 9.33333
Average turn around time = 17nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tu
s$ _
```

Program 5: Critical Section Problem

CODE:

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

#define TRUE    1
#define FALSE   0

int N;
int global = 10;
int entering[100];
int number[100];

int max(int number[100]) {
   int i = 0;
   int maximum = number[0];
   for (i = 0; i < N; i++) {
       if (maximum < number[i])
           maximum = number[i];
       }
   return maximum;
   }

void lock(int i) {
   int j = 0;
   entering[i] = TRUE;
   number[i] = 1 + max(number);
   entering[i] = FALSE;
   for (j = 0; j < N; j++) {
       while (entering[j]);
       while (number[j] != 0 && (number[j] < number[i] || (number[i] ==
number[j]) && j < i)) {}
   }
```

```
}

void unlock(int i) {
   number[i] = 0;
}

void *fn(void *integer) {
   int i = (int) integer;
   lock(i);
   printf("\n\n-----------Process %d---------",i);
   printf("\nProcess %d is Entering Critical Section\n",i);
   global++;
   printf("%d is the value of global \n",global);
   printf("Process %d is leaving Critical Section\n",i);
   printf("----------------------------------\n\n");
   unlock(i);
}


int main()
{
   printf("Enter Number of Process\n");
   scanf("%d",&N);
   int th[N];
   void *fn(void *);
   pthread_t thread[N];
   int i = 0;
   for (i = 0; i < N; i++) {
       th[i] = pthread_create(&thread[i], NULL, fn, (void *)i);
       pthread_join(thread[i], NULL);
   }
   return EXIT_SUCCESS;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ -pthread Critical_Sect
ion.cpp
Critical_Section.cpp: In function ‘int main()’:
Critical_Section.cpp:60:62: warning: cast to pointer from integer of different s
ize [-Wint-to-pointer-cast]
         th[i] = pthread_create(&thread[i], NULL, fn, (void *)i);
                                                              ^
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Enter Number of Process
2
```

Program 6: Bounded buffer problem, reader writers problem, dining philosophers problem using semaphores.

Bounded Buffer

CODE:

```
#include <boost/circular_buffer.hpp>
#include <thread>
#include <mutex>
#include <condition_variable>
#include <string>
#include <vector>
#include <iostream>

const unsigned long QUEUE_SIZE = 1000L;

using namespace std;

mutex io_mutex;
////////////////////////////////////////////////////////////////////////////////
//
//
//
////////////////////////////////////////////////////////////////////////////////
template <class T>
class BoundedBuffer
{
public:
 typedef boost::circular_buffer<T> container_type;
 typedef typename container_type::size_type  size_type;
```

```
typedef typename container_type::value_type value_type;

explicit BoundedBuffer(size_type capacity)
  : m_unread(0), m_container(capacity)
{}

BoundedBuffer(const BoundedBuffer&) = delete;

int count()
{
  return m_unread;
}

auto push_front(const value_type& item) -> bool
{
  unique_lock<mutex> lock(m_mutex);
  // 집어넣으려면 공간이 생길때까지 기다려야 한다.
  m_while_full.wait(lock,
    [this] { return m_unread < m_container.capacity() || m_shutdown; });

  if (m_shutdown)
    return false;

  m_container.push_front(item);
  ++m_unread;
  lock.unlock();
  m_while_empty.notify_all();

  return true;
}

auto pop_back(value_type* pItem) -> bool
{
  unique_lock<mutex> lock(m_mutex);

  // 빼내려면 하나이상 존재햐야 한다.
  cout << "comsumer waiting ..\n";
```

```
   m_while_empty.wait(lock, [this] { return m_unread > 0 || m_shutdown;
});

   if (m_shutdown)
     return false;

   *pItem = m_container[--m_unread];
   lock.unlock();
   m_while_full.notify_all();

   return true;
 }

 void shutdown()
 {
   m_shutdown = true;
   m_while_full.notify_all();
   m_while_empty.notify_all();
 }

private:
 bool m_shutdown {false};
 size_type m_unread;
 container_type m_container;
 mutex     m_mutex;
 condition_variable m_while_empty;
 condition_variable m_while_full;
};

template<class Buffer>
class Consumer
{
public:
 Consumer(Buffer* buffer)
   : m_container(buffer)
 {}
```

```
 void operator()()
 {
   while (true)
   {
     auto res = m_container->pop_back(&m_item);
     if (!res)
     {
       cout << "shutdown consumer\n";
       break;
     }

     {
       unique_lock<mutex> l(io_mutex);
       cout << "consumer: " << m_item << " count: " <<
m_container->count() << "\n";
     }
   }
 }

private:
 typedef typename Buffer::value_type value_type;
 Buffer* m_container;
 value_type m_item;
};

template<class Buffer>
class Producer
{
public:
 Producer(Buffer* buffer, int init)
   : m_container(buffer), m_init(init)
 {}

 void operator()()
 {
   /* TODO
   1) 쉬면서 데이타를 넣는다.
```

```
   2) 적당한 시점에 신호를 받고 종료한다.
   */

   for (int i=m_init; ; ++i)
   {
     auto res = m_container->push_front(int(i));
     if (!res)
     {
       cout << "shutdown Producer\n";
       break;
     }

     {
       unique_lock<mutex> l(io_mutex);
       cout << "producer: " << i << " count: " << m_container->count() <<
"\n";
     }

     this_thread::sleep_for(1s);
   }
 }

private:
 typedef typename Buffer::value_type value_type;
 Buffer* m_container;
 int m_init;
};

template<class Buffer>
void fifo_test(Buffer* buffer)
{
 // 1. prepare buffer
 for (int i=0; i<100; i++) buffer->push_front(i);

 // 2. prepare producers
  vector<thread> pool;
 Consumer<Buffer> consumer(buffer);
```

```
 thread consume(consumer);
 for (int i=0; i<10; i++)
   pool.emplace_back(Producer<Buffer>(buffer, i*100));

 this_thread::sleep_for(10s);

 // x. interrupts thread
 cout << "shutdwon from main thread\n";
 buffer->shutdown();
  // x. Joint the threads
 for (auto& t : pool) if (t.joinable()) t.join();
 consume.join();
}

int main(int /*argc*/, char* /*argv*/[])
{
 BoundedBuffer<int> bb_int(QUEUE_SIZE);
 fifo_test(&bb_int);

 return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ -pthread bounded_buffe
r.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
comsumer waiting ..
consumer: 0 count: 99
comsumer waiting ..
consumer: 1 count: 98
producer: 200 count: 100
producer: 300 count: 102
comsumer waiting ..
producer: 100 count: 102
producer: 0 count: 102
producer: 500 count: 103
producer: 600 count: 104
consumer: 2 count: 105
comsumer waiting ..
consumer: 3 count: 105
comsumer waiting ..
consumer: 4 count: 104
comsumer waiting ..
producer: 400 count: 104
producer: 700 count: 104
producer: 800 count: 104
consumer: 5 count: 104
```

Reader and Writer Problem

CODE:

```
#include <iostream>
#include <atomic>
#include <thread>
#include <chrono>
#include <mutex>

std::atomic_bool finish_reader(false);
std::atomic_bool finish_writer(false);



/* simple mutex option */
```

```
std::mutex mutex;
void lock_reader() {
   while (!finish_reader) {
       mutex.lock();
       std::cout  <<"I am reading" << '\n';
       mutex.unlock();
   }



}
void lock_writer() {
   while(!finish_writer) {
       mutex.lock();
       std::cout << "I am writing" << '\n';
       mutex.unlock();
   }
}

/* solution priority reader starvation writers */
std::mutex mutex_reader;
std::mutex mutex_writer;

int count_readers=0;
void multiple_lock_reader(int num_reader) {

   while (!finish_reader) {
       mutex_reader.lock();
       count_readers++;
       if (count_readers==1) {
           mutex_writer.lock();
       }
       mutex_reader.unlock();

       std::cout  <<"I am reading  reader: "<< num_reader << '\n';

       mutex_reader.lock();
       count_readers--;
```

```
        if (count_readers==0) {
            mutex_writer.unlock();
        }
        mutex_reader.unlock();
    }



}
void multiple_lock_writer() {
    while(!finish_writer) {
        mutex_writer.lock();
        std::cout << "I am writing" << '\n';
        mutex_writer.unlock();
    }

}



std::mutex x,y,z;
std::mutex rsem, wsem;

int count_readers_readers=0;
int count_readers_writers=0;
void writer_priority_lock_reader(int num_reader) {

    while (!finish_reader) {
        //other readers
        z.lock();
        //give writer priority
        rsem.lock();
        //protect counter update
        x.lock();
        count_readers_readers++;
        if (count_readers_readers==1) {
            wsem.lock();
        }
        x.unlock();
```

```
        rsem.unlock();
        z.unlock();

        std::cout  <<"I am reading  reader: "<< num_reader << '\n';

        x.lock();
        count_readers_readers--;
        if (count_readers_readers==0) {
            wsem.unlock();
        }
        x.unlock();
    }


}
void writer_priority_lock_writer() {
    while(!finish_writer) {
        y.lock();
        count_readers_writers++;
        if (count_readers_writers==1) {
            rsem.lock();
        }
        y.unlock();
        std::cout << "I am writing" << '\n';

        y.lock();
        count_readers_writers--;
        if (count_readers_writers==0) {
            rsem.unlock();
        }
        y.unlock();
    }

}


void versionLockSync() {
```

```
   std::thread tWriter(lock_writer);
   std::thread tReader(lock_reader);
   std::this_thread::sleep_for(std::chrono::seconds(4));

   finish_reader = true;
   finish_writer = true;
   tReader.join();
   tWriter.join();
}

void versionMultipleLockSync() {
   std::thread tReader(multiple_lock_reader,1);
   std::thread tReader2(multiple_lock_reader,2);
   std::thread tReader3(multiple_lock_reader,3);
   std::thread tWriter(multiple_lock_writer);
   std::this_thread::sleep_for(std::chrono::seconds(4));

   finish_reader = true;
   finish_writer = true;
   tReader.join();
   tReader2.join();
   tReader3.join();
   tWriter.join();
}









int main () {
//  versionLockSync();
   versionMultipleLockSync();
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
I am reading  reader: 1
3

I am writing
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

Dining Philosopher Problem

CODE:

```
#include<stdio.h>
#include<semaphore.h>
#include<pthread.h>
#define N 5
#define THINKING 0
#define HUNGRY 1
#define EATING 2
#define LEFT (ph_num+4)%N
#define RIGHT (ph_num+1)%N
sem_t mutex;
```

```
sem_t S[N];
void * philospher(void *num);
void take_fork(int);
void put_fork(int);
void test(int);
int state[N];
int phil_num[N]={0,1,2,3,4};
int main()
{
  int i;
  pthread_t thread_id[N];
  sem_init(&mutex,0,1);
  for(i=0;i<N;i++)
      sem_init(&S[i],0,0);
  for(i=0;i<N;i++)
  {
      pthread_create(&thread_id[i],NULL,philospher,&phil_num[i]);
      printf("Philosopher %d is thinkingn",i+1);
  }
  for(i=0;i<N;i++)
      pthread_join(thread_id[i],NULL);
}
void *philospher(void *num)
{
  while(1)
  {
      int *i = num;
      sleep(1);
      take_fork(*i);
      sleep(0);
      put_fork(*i);
  }
}
void take_fork(int ph_num)
{
  sem_wait(&mutex);
  state[ph_num] = HUNGRY;
```

```
  printf("Philosopher %d is Hungryn",ph_num+1);
  test(ph_num);
  sem_post(&mutex);
  sem_wait(&S[ph_num]);
  sleep(1);
}
void test(int ph_num)
{
  if (state[ph_num] == HUNGRY && state[LEFT] != EATING && state[RIGHT] !=
EATING)
  {
      state[ph_num] = EATING;
      sleep(2);
      printf("Philosopher %d takes fork %d and
%dn",ph_num+1,LEFT+1,ph_num+1);
      printf("Philosopher %d is Eatingn",ph_num+1);
      sem_post(&S[ph_num]);
  }
}
void put_fork(int ph_num)
{
  sem_wait(&mutex);
  state[ph_num] = THINKING;
  printf("Philosopher %d putting fork %d and %d
downn",ph_num+1,LEFT+1,ph_num+1);
  printf("Philosopher %d is thinkingn",ph_num+1);
  test(LEFT);
  test(RIGHT);
  sem_post(&mutex);
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts/dining-philosophers
File  Edit  View  Search  Terminal  Help
Unpacking objects: 100% (14/14), done.
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ cd dining-philosophers/
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts/dining-philosophers$ gcc di
ningPhilosophersSemaphores.c -lpthread && ./a.out 6
diningPhilosophersSemaphores.c: In function ‘philosopher’:
diningPhilosophersSemaphores.c:30:3: warning: implicit declaration of function ‘
sleep’ [-Wimplicit-function-declaration]
   sleep(rand()%10+1);
   ^~~~~
 T -  T -  T -  T -  T -  T -
 T -  T -  T -  T -  T -  T -
 T -  T -  T -  T -  T -  T -
 E -  T -  T -  T -  E -  T -
 E -  T -  T -  T -  E -  T -
 E -  T -  T -  H -  E -  H -
 E -  T -  H -  E -  T -  H -
 E -  H -  H -  E -  T -  H -
 E -  H -  H -  T -  T -  H -
 E -  H -  H -  T -  T -  H -
 T -  H -  E -  T -  T -  E -
 T -  H -  E -  H -  T -  T -
 T -  H -  E -  H -  T -  T -
^C
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts/dining-philosophers$
```

Problem 7: Banker’s Algorithm

CODE:

```
#include <iostream>
using namespace std;
int n, m, i, j, k;

void process(){

n = 5;
   m = 3;
   int allocation[5][3] = { { 0, 1, 0 },
                            { 2, 0, 0 },
                            { 3, 0, 2 },
                            { 2, 1, 1 },
                            { 0, 0, 2 } };

   int maximum[5][3] = { { 7, 5, 3 },
                         { 3, 2, 2 },
                         { 9, 0, 2 },
                         { 2, 2, 2 },
                         { 4, 3, 3 } };

   int available[3] = { 3, 3, 2 };

   int fun[n], answer[n], index = 0;
   for (k = 0; k < n; k++) {
       fun[k] = 0;
   }
   int need[n][m];
   for (i = 0; i < n; i++) {
       for (j = 0; j < m; j++)
           need[i][j] = maximum[i][j] - allocation[i][j];
   }
```

```
    int y = 0;
    for (k = 0; k < 5; k++) {
        for (i = 0; i < n; i++) {
            if (fun[i] == 0) {

                int flag = 0;
                for (j = 0; j < m; j++) {
                    if (need[i][j] > available[j]){
                        flag = 1;
                        break;
                    }
                }

                if (flag == 0) {
                    answer[index++] = i;
                    for (y = 0; y < m; y++)
                        available[y] += allocation[i][y];
                    fun[i] = 1;
                }
            }
        }
    }

    cout << "the safe sequence is as following " << endl;
    for (i = 0; i < n - 1; i++)
        cout << " P" << answer[i] << " ->";
    cout << " P" << answer[n - 1] <<endl;

}
int main()
{

    process();



    return (0);
```

```
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
the safe sequence is as following
 P1 -> P3 -> P4 -> P0 -> P2
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

Problem 8: Page Replacement Algorithms

Least Recently Used

CODE:

```
#include<bits/stdc++.h>
using namespace std;


int pgFaults(int p[], int n, int c)
{

   set<int> s;




   //queue<int> ind;


   int ans = 0;
   for (int i=0; i<n; i++)
   {

       if (s.size() < c)
       {

           if (s.find(p[i])==s.end())
           {

               s.insert(p[i]);


               ans++;
```

```
            //ind.push(p[i]);
        }
    }


    else
    {

        if (s.find(p[i]) == s.end())
        {

            //int val = ind.front();

            set<int> :: iterator itr;

            int val = -1;
            int mini = 1000;

            for(itr = s.begin();itr != s.end();itr++){

                int temp = *itr;

                int j ;
                for ( j = i-1;j>=0;j--){

                    if( p[j]==temp)
                        break;
                }
                if( j<mini){

                    mini = j ;
                    val= temp;
                }
```

```
                }

                //ind.pop();

                s.erase(val);

                s.insert(p[i]);

                //ind.push(p[i]);

                ans++;
            }
        }
    }

    return ans;
}


int main()
{
    int pag[] = {7, 0, 1, 2, 0, 3, 0, 4,
                 2, 3, 0, 3, 2};
    int siz = sizeof(pag)/sizeof(pag[0]);
    int cap = 4;
    cout << pgFaults(pag, siz, cap);
    return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ page_replacement_lru.c
pp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
6nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

First In First Out

CODE:

```
#include<bits/stdc++.h>
using namespace std;


int pgFaults(int p[], int n, int c)
{

   unordered_set<int> s;


   queue<int> ind;


   int ans = 0;
   for (int i=0; i<n; i++)
   {

       if (s.size() < c)
       {

           if (s.find(p[i])==s.end())
           {

               s.insert(p[i]);
```

```
                ans++;


                ind.push(p[i]);
            }
        }


        else
        {

            if (s.find(p[i]) == s.end())
            {

                int val = ind.front();


                ind.pop();


                s.erase(val);


                s.insert(p[i]);


                ind.push(p[i]);


                ans++;
            }
        }
    }

    return ans;
}
```

```
int main()
{
   int pag[] = {7, 0, 1, 2, 0, 3, 0, 4,
                2, 3, 0, 3, 2};
   int siz = sizeof(pag)/sizeof(pag[0]);
   int cap = 4;
   cout << pgFaults(pag, siz, cap);
   return 0;
}
```

OUTPUT:

Optimal Page Replacement

CODE:

```
#include<bits/stdc++.h>
using namespace std;


int pgFaults(int p[], int n, int c)
{

   set<int> s;




   //queue<int> ind;
```

```
int ans = 0;
for (int i=0; i<n; i++)
{

    if (s.size() < c)
    {

        if (s.find(p[i])==s.end())
        {

            s.insert(p[i]);


            ans++;


            //ind.push(p[i]);
        }
    }


    else
    {

        if (s.find(p[i]) == s.end())
        {

            //int val = ind.front();

            set<int> :: iterator itr;

            int val = -1;
            int maxi = 0;

            for(itr = s.begin();itr != s.end();itr++){
```

```
                int temp = *itr;

                int j ;
                for ( j = i+1;j<n;j++){

                    if( p[j]==temp)
                        break;
                }
                if( j>maxi){

                    maxi = j ;
                    val= temp;
                }


            }


            //ind.pop();


            s.erase(val);


            s.insert(p[i]);


            //ind.push(p[i]);


            ans++;
        }
    }
}

return ans;
```

```
}



int main()
{
   int pag[] = {7, 0, 1, 2, 0, 3, 0, 4,
                2, 3, 0, 3, 2};
   int siz = sizeof(pag)/sizeof(pag[0]);
   int cap = 4;
   cout << pgFaults(pag, siz, cap)<<endl;
   return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ page_replacement_optim
al.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
6
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

Program 9: Threads

CODE:

```
#include <iostream>
#include <thread>
using namespace std;


void dummy(int x)
{
   for (int i = 0; i < x; i++) {
       cout << "Thread using function"
           " pointer \n";
   }
}


class t_object {
public:
   void operator()(int x)
   {
       for (int i = 0; i < x; i++)
           cout << "Thread using function"
               " object \n";
   }
};

int main()
{
   cout << "Programme to demonstrate multi-threading by using 3 threads
-Threads 1 and 2 and 3 " << endl;


   thread thread1(dummy, 5);
```

```
    thread thread2(t_object(), 5);


    auto frie = [](int x) {
        for (int i = 0; i < x; i++)
            cout << "Thread using lambda"
            " expression \n";
    };


    thread thread3(frie, 5);


    thread1.join();


    thread2.join();


    thread3.join();

    return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ -pthread Threads.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Programme to demonstrate multi-threading by using 3 threads -Threads 1 and 2 and
 3
Thread using function pointer
Thread using function pointer
Thread using function pointer
Thread using function pointer
Thread using function pointer
Thread using function object
Thread using function object
Thread using function object
Thread using function object
Thread using function object
Thread using lambda expression
Thread using lambda expression
Thread using lambda expression
Thread using lambda expression
Thread using lambda expression
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

Program 10: File Manipulation

open()

CODE:

```
#include<stdio.h>
#include<fcntl.h>
#include<errno.h>
extern int errno;
int main()
{

   int file_descripter = open("sample.txt", O_RDONLY | O_CREAT);

   printf("file descripter = %d\n", file_descripter);

   if (file_descripter ==-1)
   {
       printf("Error Number % d\n", errno);

       perror("Program");
   }
   return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ open_system_call.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
file descripter = 3
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

close()

CODE:

```
#include<stdio.h>
#include <fcntl.h>
#include<iostream>
#include<stdlib.h>
#include <unistd.h>
int main()
{
   int file_descripter_1 = open("os.txt", O_RDONLY| O_CREAT);
   if (file_descripter_1 < 0)
   {
       perror("c1");
```

```
        exit(1);
    }
    printf("recently opened fd = % d\n", file_descripter_1);


    if (close(file_descripter_1) < 0)
    {
        perror("c1");
        exit(1);
    }
    printf("closed this fd.\n");
    printf("Now opening another file descripter as the previous one is
pointing to NULL \n");


    int file_descripter_2 = open("sample.txt", O_RDONLY, 0);

    printf("fd2 = % d\n", file_descripter_2);
    exit(0);
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ close_system_call\ .cp
p
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
recently opened fd =  3
closed this fd.
Now opening another file descripter as the previous one is pointing to NULL
fd2 = -1
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

read()

CODE:

```
#include<stdlib.h>
#include<stdio.h>
#include <fcntl.h>
#include <unistd.h>
int main()
{
int file_descripter, siz;
char *str = (char *) calloc(100, sizeof(char));

file_descripter = open("os.txt", O_RDONLY);
if (file_descripter < 0) { perror("r1"); exit(1); }
```

```
siz = read(file_descripter, str, 10);
printf("called read(% d, c, 10). returned that"
       " %d bytes were read.\n", file_descripter, siz);
str[siz] = '\0';
printf("Those bytes are as follows: % s\n", str);
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ read_system_call.cpp
read_system_call.cpp: In function ‘int main()’:
read_system_call.cpp:17:48: warning: ' ' flag used with ‘%s’ gnu_printf format [
-Wformat=]
 printf("Those bytes are as follows: % s\n", str);
                                                ^
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
called read( 3, c, 10). returned that 10 bytes were read.
Those bytes are as follows: sample tex
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

write()

CODE:

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

```
#include<stdio.h>
#include <fcntl.h>
#include<string>
#include<cstring>
#include <unistd.h>
main()
{
int siz;

int file_descripter = open("sample.txt", O_WRONLY | O_CREAT | O_TRUNC,
0644);
if (file_descripter < 0)
{
   perror("r1");
   exit(1);
}

siz = write(file_descripter, "hello world\n", strlen("hello world\n"));

printf("called write(% d, \"hello world\\n\", %d)."
   " It returned %d\n", file_descripter, strlen("hello world\n"), siz);

close(file_descripter);
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ write_system_call.cpp
write_system_call.cpp: In function ‘int main()’:
write_system_call.cpp:21:68: warning: format ‘%d’ expects argument of type ‘int’
, but argument 3 has type ‘size_t {aka long unsigned int}’ [-Wformat=]
  " It returned %d\n", file_descripter, strlen("hello world\n"), siz);
                                                                    ^
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
called write( 3, "hello world\n", 12). It returned 12
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

Program 11: Disk Scheduling Algorithms

First Come First Serve

CODE:

```
#include <iostream>

using namespace std;
 int size = 8;
 void FCFS(int arr[], int head)
{
   int seek_count = 0;
   int distance, cur_track;
    for (int i = 0; i < size; i++)
   {
       cur_track = arr[i];
        distance = abs(cur_track - head);

       seek_count += distance;
        head = cur_track;
   }
    cout << "Total number of seek operations = " << seek_count << endl;
    cout << "Seek Sequence is" << endl;
    for (int i = 0; i < size; i++)
   {
       cout << arr[i] << endl;
   }
}

int main()
{
   int arr[size] = { 176, 79, 34, 60, 92, 11, 41, 114 };
   int head = 50;
    FCFS(arr, head);
    return 0;
```

```
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ Disk_FCFS.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Total number of seek operations = 510
Seek Sequence is
176
79
34
60
92
11
41
114
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```

C-SCAN

CODE:

```
#include <iostream>
#include<bits/stdc++.h>
using namespace std;
 int size = 8;
int disk_size = 200;
 void CSCAN(int arr[], int head)
{
   int seek_count = 0;
   int distance, cur_track;
   std::vector<int> left, right;
```

```
vector<int> seek_sequence;
 left.push_back(0);
right.push_back(disk_size - 1);
 for (int i = 0; i < size; i++)
{
    if (arr[i] < head)
        left.push_back(arr[i]);
    if (arr[i] > head)
        right.push_back(arr[i]);
}
 std::sort(left.begin(), left.end());
std::sort(right.begin(), right.end());

for (int i = 0; i < right.size(); i++)
{
    cur_track = right[i];

    seek_sequence.push_back(cur_track);
     distance = abs(cur_track - head);
     seek_count += distance;
     head = cur_track;
}
 head = 0;
 for (int i = 0; i < left.size(); i++)
{
    cur_track = left[i];
     seek_sequence.push_back(cur_track);
     distance = abs(cur_track - head);
     seek_count += distance;
     head = cur_track;
}
 cout << "Total number of seek operations = "  << seek_count << endl;
 cout << "Seek Sequence is" << endl;
 for (int i = 0; i < seek_sequence.size(); i++)
{
    cout << seek_sequence[i] << endl;
}
```

```
}
 int main()
{
   int arr[size] = { 176, 79, 34, 60, 92, 11, 41, 114 };
   int head = 50;
    cout << "Initial position of head: " << head << endl;
   CSCAN(arr, head);
    return 0;
}
```

OUTPUT:

```
nipunika@nipunika-Nitro-AN515-52: ~/Downloads/OS-tuts
File  Edit  View  Search  Terminal  Help
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ g++ Disk_C_SCAN.cpp
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$ ./a.out
Initial position of head: 50
Total number of seek operations = 190
Seek Sequence is
60
79
92
114
176
199
0
11
34
41
nipunika@nipunika-Nitro-AN515-52:~/Downloads/OS-tuts$
```