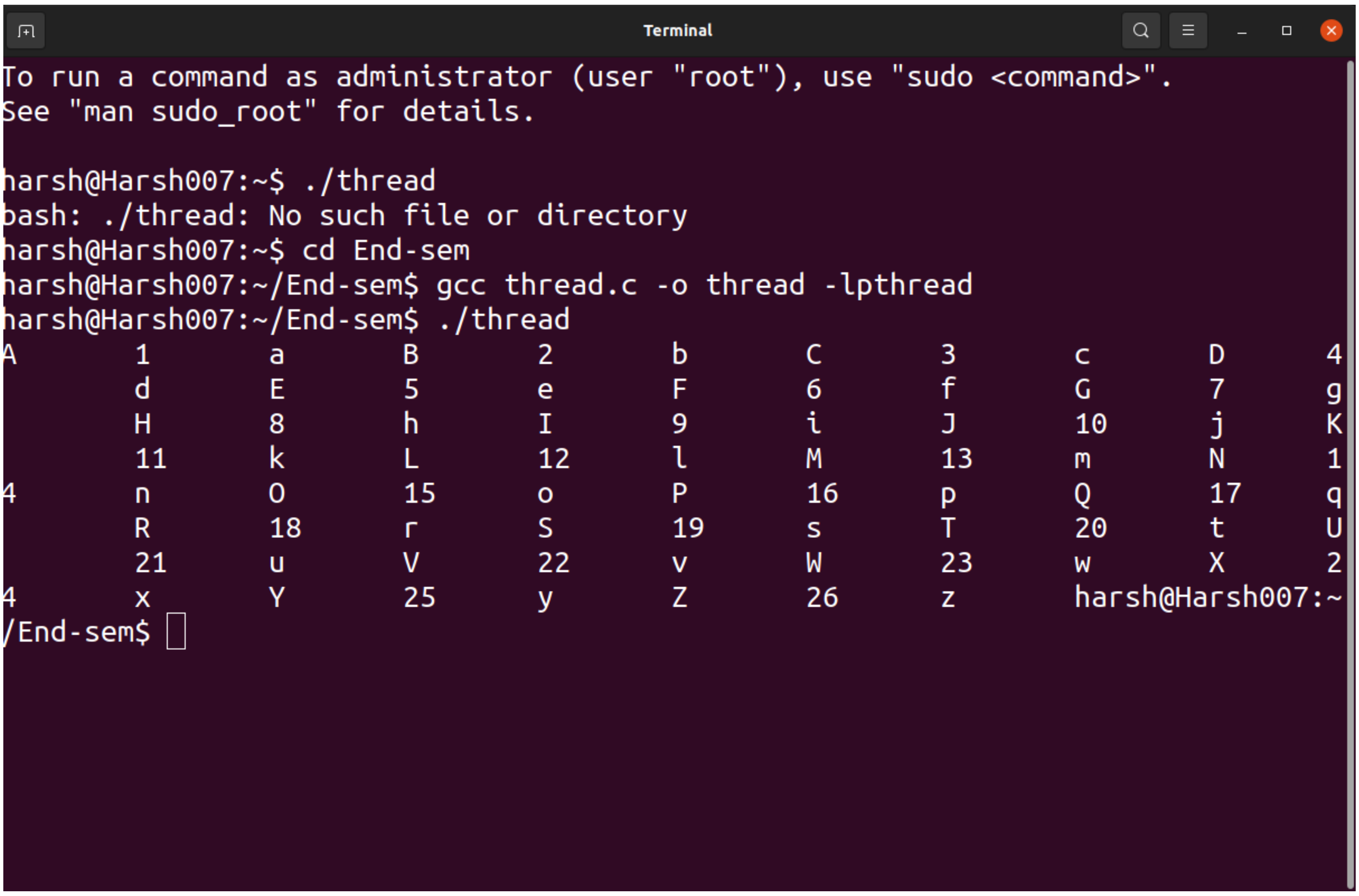
1 -B) Multithreads :-

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

int num = 0;
pthread_mutex_t mutex;

void* task() {
    while(num<25) {
        pthread_mutex_lock(&mutex);
        num++;
        printf("%c \t", num+64);
        printf("%d \t", num);
        printf("%c \t", num+96);
        pthread_mutex_unlock(&mutex);

    }
}

int main(int argc, char* argv[]) {
    pthread_t p1, p2, p3;
    pthread_mutex_init(&mutex, NULL);
    if (pthread_create(&p1, NULL, &task, NULL) != 0) {
        return 1;
    }
    if (pthread_create(&p2, NULL, &task, NULL) != 0) {
        return 2;
    }
    if (pthread_create(&p3, NULL, &task, NULL) != 0) {
        return 3;
    }

    if (pthread_join(p1, NULL) != 0) {
        return 5;
    }
    if (pthread_join(p2, NULL) != 0) {
        return 6;
    }
    if (pthread_join(p3, NULL) != 0) {
        return 7;
    }

    pthread_mutex_destroy(&mutex);

    return 0;
}
```



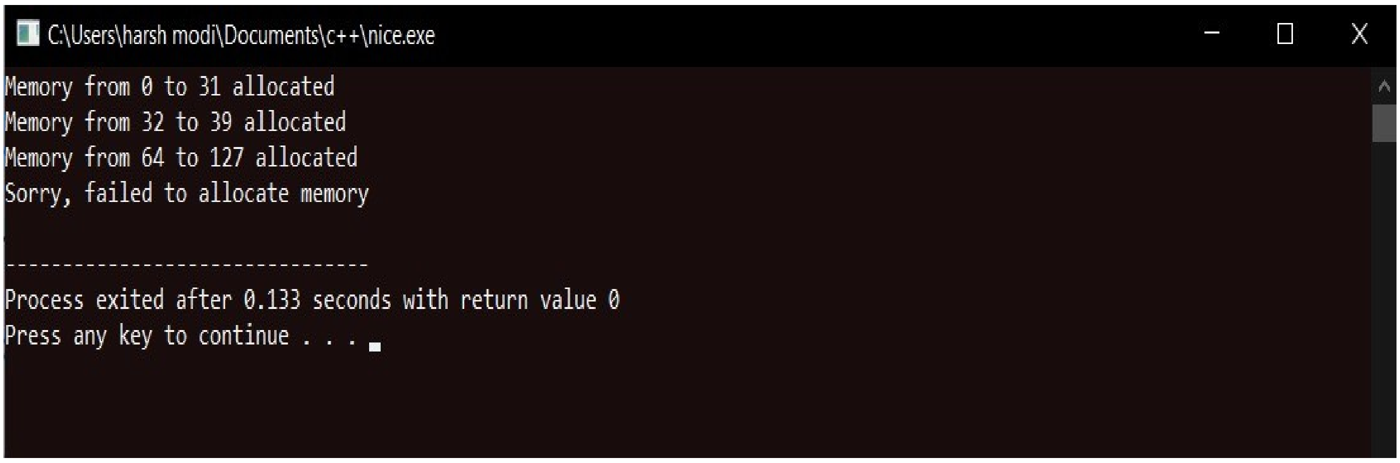
```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

harsh@Harsh007:~$ ./thread
bash: ./thread: No such file or directory
harsh@Harsh007:~$ cd End-sem
harsh@Harsh007:~/End-sem$ gcc thread.c -o thread -lpthread
harsh@Harsh007:~/End-sem$ ./thread
A       1       a       B       2       b       C       3       c       D      4
        d       E       5       e       F       6       f       G       7      g
        H       8       h       I       9       i       J       10      j      K
        11      k       L       12      l       M       13      m       N      1
4       n       O       15      o       P       16      p       Q       17     q
        R       18      r       S       19      s       T       20      t      U
        21      u       V       22      v       W       23      w       X      2
4       x       Y       25      y       Z       26      z       harsh@Harsh007:~
/End-sem$ 
```

2 – B)

--> Memory allocation (code)

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


int size;

vector<pair<int, int> > count[100000];

map<int, int> mp;

void initialize(int sz)
{

    int n = ceil(log(sz) / log(2));
    size = n + 1;

    for(int i = 0; i <= n; i++)
        count[i].clear();

    count[n].push_back(make_pair(0, sz - 1));
}
```

```
void allocate(int sz)
{

    int n = ceil(log(sz) / log(2));


    if (count[n].size() > 0)
    {
        pair<int, int> temp = count[n][0];


        count[n].erase(count[n].begin());
        cout << "Memory from " << temp.first
             << " to " << temp.second << " allocated"
             << "\n";

        mp[temp.first] = temp.second -
                         temp.first + 1;
    }
    else
    {
        int i;
        for(i = n + 1; i < size; i++)
        {


            if(count[i].size() != 0)
                break;
        }

        if (i == size)
        {
            cout << "Sorry, failed to allocate memory \n";
        }

        else
        {
            pair<int, int> temp;
            temp = count[i][0];


            count[i].erase(count[i].begin());
            i--;

            for(; i >= n; i--)
            {

                pair<int, int> pair1, pair2;
                pair1 = make_pair(temp.first,
                                  temp.first +
                                  (temp.second -
                    temp.first) / 2);
            pair2 = make_pair(temp.first +
                    (temp.second -
                    temp.first + 1) / 2,
                    temp.second);

            count[i].push_back(pair1);


            count[i].push_back(pair2);
            temp = count[i][0];


            count[i].erase(count[i].begin());
          }
          cout << "Memory from " << temp.first
             << " to " << temp.second
             << " allocated" << "\n";

          mp[temp.first] = temp.second -
                    temp.first + 1;
        }
    }
}


int main()
{


    initialize(128);
    allocate(32);
    allocate(7);
    allocate(64);
    allocate(56);

    return 0;
}
```



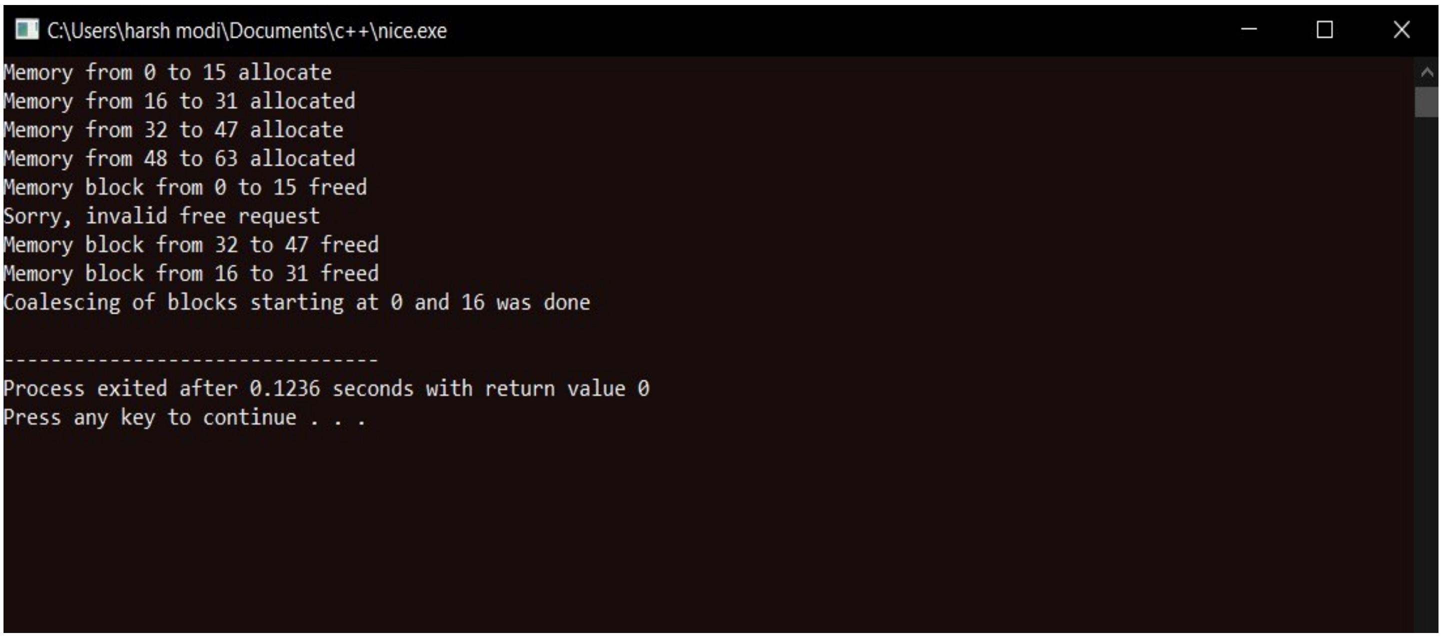
```
Memory from 0 to 31 allocated
Memory from 32 to 39 allocated
Memory from 64 to 127 allocated
Sorry, failed to allocate memory

--------------------------------
Process exited after 0.133 seconds with return value 0
Press any key to continue . . .
```

--> Memory dealocation (code)

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

int size;

vector<pair<int, int> > new_list[100000];


map<int, int> mp;

void Buddy(int s)
{

    int n = ceil(log(s) / log(2));

    size = n + 1;
    for(int i = 0; i <= n; i++)
        new_list[i].clear();


    new_list[n].push_back(make_pair(0, s - 1));
}

void m_alloc(int s)
{


    int x = ceil(log(s) / log(2));


    if (new_list[x].size() > 0)
    {
        pair<int, int> temp = new_list[x][0];


        new_list[x].erase(new_list[x].begin());

        cout << "Memory from " << temp.first
             << " to " << temp.second
             << " allocated" << "\n";


        mp[temp.first] = temp.second -
                         temp.first + 1;
    }
    else
    {
        int i;
```

```
for(i = x + 1; i < size; i++)
{


    if (new_list[i].size() != 0)
        break;
}


if (i == size)
{
    cout << "Sorry, failed to allocate memory\n";
}


else
{
    pair<int, int> temp;
    temp = new_list[i][0];


    new_list[i].erase(new_list[i].begin());
    i--;

    for(;i >= x; i--)
    {


        pair<int, int> pair1, pair2;
        pair1 = make_pair(temp.first,
                          temp.first +
                         (temp.second -
                          temp.first) / 2);
        pair2 = make_pair(temp.first +
                         (temp.second -
                          temp.first + 1) / 2,
                          temp.second);

        new_list[i].push_back(pair1);


        new_list[i].push_back(pair2);
        temp = new_list[i][0];


        new_list[i].erase(new_list[i].begin());
    }

    cout << "Memory from " << temp.first
         << " to " << temp.second
         << " allocate" << "\n";
```

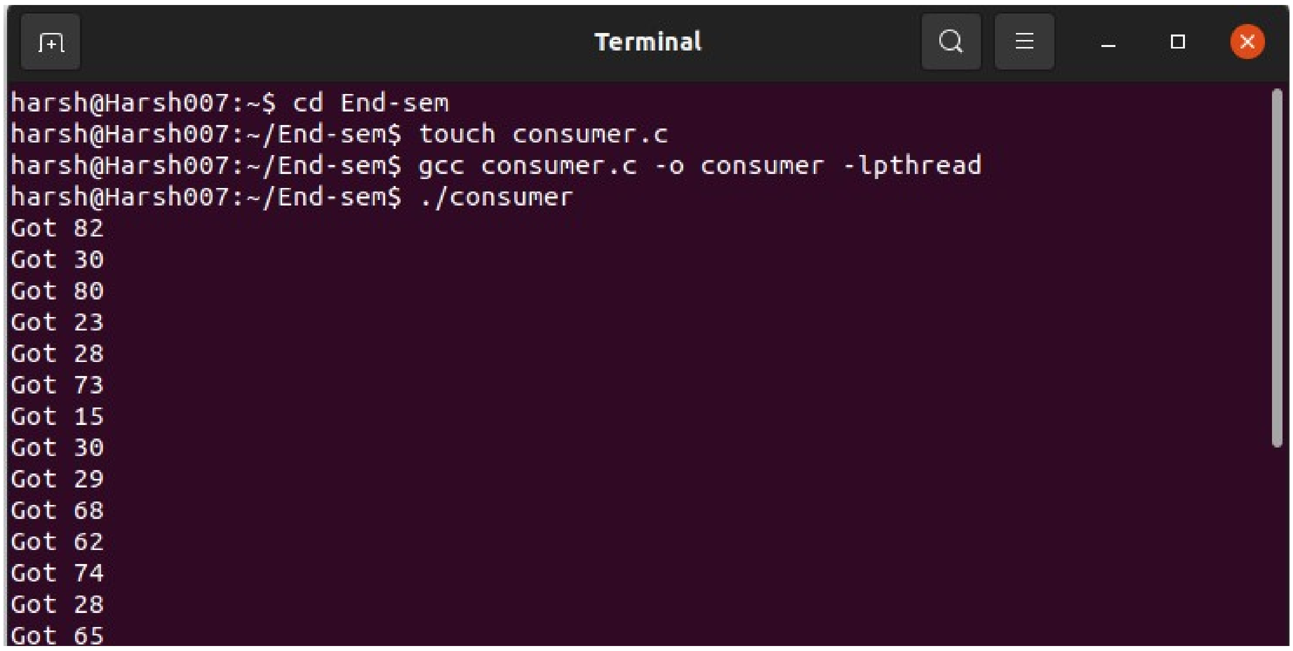
```
            mp[temp.first] = temp.second -
                        temp.first + 1;
        }
    }
}

void d_alloc(int id)
{

    if(mp.find(id) == mp.end())
    {
        cout << "Sorry, invalid free request\n";
        return;
    }


    int n = ceil(log(mp[id]) / log(2));

    int i, buddyNumber, buddyAddress;


    new_list[n].push_back(make_pair(id,
                          id + pow(2, n) - 1));
    cout << "Memory block from " << id
         << " to "<< id + pow(2, n) - 1
         << " freed\n";


    buddyNumber = id / mp[id];

    if (buddyNumber % 2 != 0)
        buddyAddress = id - pow(2, n);
    else
        buddyAddress = id + pow(2, n);


    for(i = 0; i < new_list[n].size(); i++)
    {


        if (new_list[n][i].first == buddyAddress)
        {


            if (buddyNumber % 2 == 0)
            {
                new_list[n + 1].push_back(make_pair(id,
                   id + 2 * (pow(2, n) - 1)));

                cout << "Coalescing of blocks starting at "
                     << id << " and " << buddyAddress
                << " was done" << "\n";
            }
            else
            {
                new_list[n + 1].push_back(make_pair(
                    buddyAddress, buddyAddress +
                    2 * (pow(2, n))));

                cout << "Coalescing of blocks starting at "
                     << buddyAddress << " and "
                     << id << " was done" << "\n";
            }
            new_list[n].erase(new_list[n].begin() + i);
            new_list[n].erase(new_list[n].begin() +
            new_list[n].size() - 1);
            break;
        }
    }

    mp.erase(id);
}


int main()
{

    Buddy(128);
    m_alloc(16);
    m_alloc(16);
    m_alloc(16);
    m_alloc(16);
    d_alloc(0);
    d_alloc(9);
    d_alloc(32);
    d_alloc(16);

    return 0;
}
```


C:\Users\harsh modi\Documents\c++\nice.exe

```
Memory from 0 to 15 allocate
Memory from 16 to 31 allocated
Memory from 32 to 47 allocate
Memory from 48 to 63 allocated
Memory block from 0 to 15 freed
Sorry, invalid free request
Memory block from 32 to 47 freed
Memory block from 16 to 31 freed
Coalescing of blocks starting at 0 and 16 was done

--------------------------------
Process exited after 0.1236 seconds with return value 0
Press any key to continue . . .
```

3)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>
#include <unistd.h>
#include <time.h>
#include <semaphore.h>

#define THREAD_NUM 8

sem_t semEmpty;
sem_t semFull;

pthread_mutex_t mutexBuffer;

int buffer[10];
int count = 0;

void* producer(void* args) {
    while (1) {
        // Produce
        int x = rand() % 100;
        sleep(2);

        // Add to the buffer
        sem_wait(&semEmpty);
        pthread_mutex_lock(&mutexBuffer);
        buffer[count] = x;
        count++;
        pthread_mutex_unlock(&mutexBuffer);
        sem_post(&semFull);
    }
}

void* consumer(void* args) {
    while (1) {
        int y;

        // Remove from the buffer
        sem_wait(&semFull);
        pthread_mutex_lock(&mutexBuffer);
        y = buffer[count - 1];
        count--;
        pthread_mutex_unlock(&mutexBuffer);
        sem_post(&semEmpty);

        // Consume
        printf("Got %d\n", y);
        sleep(1);
    }
}

int main(int argc, char* argv[]) {
    srand(time(NULL));
    pthread_t th[THREAD_NUM];
    pthread_mutex_init(&mutexBuffer, NULL);
    sem_init(&semEmpty, 0, 10);
    sem_init(&semFull, 0, 0);
    int i;
    for (i = 0; i < THREAD_NUM; i++) {
        if (i > 0) {
            if (pthread_create(&th[i], NULL, &producer, NULL) != 0) {
                perror("Failed to create thread");
            }
        } else {
            if (pthread_create(&th[i], NULL, &consumer, NULL) != 0) {
                perror("Failed to create thread");
            }
        }
    }
    for (i = 0; i < THREAD_NUM; i++) {
        if (pthread_join(th[i], NULL) != 0) {
            perror("Failed to join thread");
        }
    }
    sem_destroy(&semEmpty);
    sem_destroy(&semFull);
    pthread_mutex_destroy(&mutexBuffer);
    return 0;
}
```



```
harsh@Harsh007:~$ cd End-sem
harsh@Harsh007:~/End-sem$ touch consumer.c
harsh@Harsh007:~/End-sem$ gcc consumer.c -o consumer -lpthread
harsh@Harsh007:~/End-sem$ ./consumer
Got 82
Got 30
Got 80
Got 23
Got 28
Got 73
Got 15
Got 30
Got 29
Got 68
Got 62
Got 74
Got 28
Got 65
```