OS LAB 9

NAME: Aditya Anand

ROLL NO.: 20124009

BRANCH: IT

|  |  |  |  |
| --- | --- | --- | --- |
| S No. | Title | Date of Implementation | Remarks |
| 1 | Implementation of First In First Out Page Replacement Algorithm | 18-04-2022 |  |
| 2 | Implementation of First In First Out Page Replacement Algorithm | 18-04-2022 |  |
| 3 | Implementation of First In First Out Page Replacement Algorithm | 18-04-2022 |  |
| 4 | Simulation Of Paging Techniques In Memory Management | 18-04-2022 |  |

IMPLEMENTATION OF FIRST IN FIRST OUT PAGE REPLACEMENT ALGORITHM

CODE:

#include<bits/stdc++.h>

using namespace :: std;

int search(vector<int> arr, int key){

    for(int i=0; i<arr.size(); i++){

        if(arr[i]==key){

            return i;

        }

    }

    return -1;

}

void FIFO(vector<int> ref, int frames){

    vector<int> fr(frames, -1);

    int k = 0;

    int hit = 0, miss = 0;

    for(int i=0; i<ref.size(); i++){

        int id = search(fr, ref[i]);

        if(id!=-1){

            hit++;

        }

        else{

            fr[k]=ref[i];

            k=(k+1)%frames;

            miss++;

        }

    }

    cout<<"Hit Percentage: "<<(100.0\*hit)/(1.0\*(hit+miss))<<"%\n";

    cout<<"Miss Percentage: "<<(100.0\*miss)/(1.0\*(hit+miss))<<"%\n";

}

int main(){

    cout << "FIRST IN FIRST OUT PAGE REPLACEMENT ALGORITHM C++ IMPLEMENTATION\n";

    cout << "Name: Aditya Anand\tRoll No.:20124009\t Branch: IT\n\n\n";

    int frames = 0;

    cout<<"Enter the number of frames: ";

    cin>>frames;

    if(frames<1){

        cout<<"No frames available!";

        return 0;

    }

    cout<<"Enter the size of the reference string: ";

    int n = 0;

    cin>>n;

    cout<<"Enter the order in which pages are accessed by the CPU\n";

    vector<int> ref(n, 0);

    for(int i=0; i<n; i++){

        cin>>ref[i];

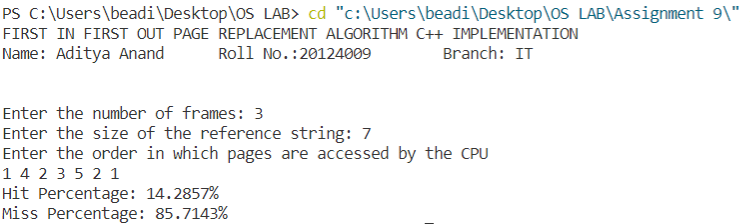
    }

    FIFO(ref, frames);

    return 0;

}

RESULT:



IMPLEMENTATION OF LEAST RECENTLY USED PAGE REPLACEMENT ALGORITHM

CODE:

#include<bits/stdc++.h>

using namespace :: std;

struct frame{

    int frNo, lastOcc;

};

// returns the index at which the key is found and the index of the least recently used page

pair<int, int> Search(vector<frame> &arr, int key, vector<int> ref, int cur){

    int id = -1, lru = INT\_MAX;

    for(int i=0; i<arr.size(); i++){

        if(arr[i].frNo==key){

            id = i;

        }

        for(int j=cur-1; j>=0; j--){

            if(ref[j]==arr[i].frNo){

                arr[i].lastOcc = j;

                lru = min(lru, arr[i].lastOcc);

                break;

            }

        }

    }

    int pRep = -1;

    for(int i=0; i<arr.size(); i++){

        if(arr[i].lastOcc == lru){

            pRep = i;

            break;

        }

    }

    return make\_pair(id, pRep);

}

void LRU(vector<int> ref, int frames){

    vector<frame> fr(frames);

    for(int i=0; i<frames; i++){

        fr[i].frNo = -1;

        fr[i].lastOcc = -1;

    }

    int hit = 0, miss = 0;

    int k=0, it=0;

    while(k<frames && it<ref.size()){

        bool found = false;

        for(int j=k; j>=0; j--){

            if(fr[j].frNo == ref[it]){

                found=true;

                hit++;

                break;

            }

        }

        if(!found){

            fr[k].frNo = ref[it];

            fr[k].lastOcc = it;

            miss++;

            k++;

        }

        it++;

    }

    for(int i=it; i<ref.size(); i++){

        pair<int , int> p = Search(fr, ref[i], ref, i);

        if(p.first!=-1){

            hit++;

        }

        else{

            fr[p.second].frNo=ref[i];

            fr[p.second].lastOcc=i;

            miss++;

        }

    }

    cout<<"Hit Percentage: "<<(100.0\*hit)/(1.0\*(hit+miss))<<"%\n";

    cout<<"Miss Percentage: "<<(100.0\*miss)/(1.0\*(hit+miss))<<"%\n";

}

int main(){

    cout << "LEAST RECENTLY USED REPLACEMENT ALGORITHM C++ IMPLEMENTATION\n";

    cout << "Name: Aditya Anand\tRoll No.:20124009\t Branch: IT\n\n\n";

    int frames = 0;

    cout<<"Enter the number of frames: ";

    cin>>frames;

    if(frames<1){

        cout<<"No frames available!";

        return 0;

    }

    cout<<"Enter the size of the reference string: ";

    int n = 0;

    cin>>n;

    cout<<"Enter the order in which pages are accessed by the CPU\n";

    vector<int> ref(n);

    for(int i=0; i<n; i++){

        cin>>ref[i];

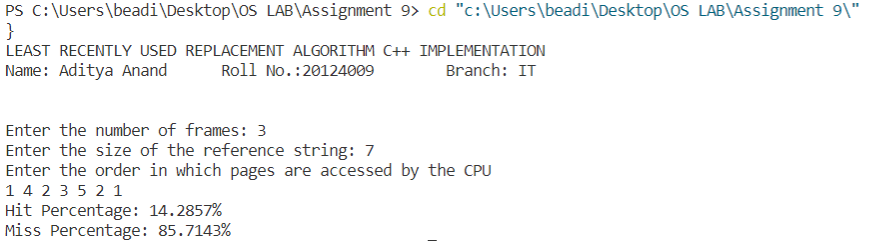
    }

    LRU(ref, frames);

    return 0;

}

RESULT:



IMPLEMENTATION OF LEAST FREQUENTLY USED PAGE REPLACEMENT ALGORITHM

CODE:

#include<bits/stdc++.h>

using namespace :: std;

struct frame{

    int frNo, freq;

};

// returns the index at which the key is found and the index of the least frequently used page

pair<int, int> Search(vector<frame> &arr, int key, vector<int> ref, int cur){

    int id = -1, lfu = INT\_MIN;

    for(int i=0; i<arr.size(); i++){

        if(arr[i].frNo==key){

            id = i;

        }

        lfu = max(lfu, arr[i].freq);

    }

    int pRep = -1;

    for(int i=0; i<arr.size(); i++){

        if(arr[i].freq == lfu){

            pRep = i;

            break;

        }

    }

    return make\_pair(id, pRep);

}

void LRU(vector<int> ref, int frames){

    vector<frame> fr(frames);

    for(int i=0; i<frames; i++){

        fr[i].frNo = -1;

        fr[i].freq = 0;

    }

    int hit = 0, miss = 0;

    int k=0, it=0;

    while(k<frames && it<ref.size()){

        bool found = false;

        for(int j=k; j>=0; j--){

            if(fr[j].frNo == ref[it]){

                found=true;

                fr[j].freq++;

                hit++;

                break;

            }

        }

        if(!found){

            fr[k].frNo = ref[it];

            fr[k].freq=1;

            miss++;

            k++;

        }

        it++;

    }

    for(int i=it; i<ref.size(); i++){

        pair<int , int> p = Search(fr, ref[i], ref, i);

        if(p.first!=-1){

            fr[p.first].freq++;

            hit++;

        }

        else{

            fr[p.second].frNo=ref[i];

            fr[p.second].freq=1;

            miss++;

        }

    }

    cout<<"Hit Percentage: "<<(100.0\*hit)/(1.0\*(hit+miss))<<"%\n";

    cout<<"Miss Percentage: "<<(100.0\*miss)/(1.0\*(hit+miss))<<"%\n";

}

int main(){

    cout << "LEAST FREQUENTLY USED PAGE REPLACEMENT ALGORITHM C++ IMPLEMENTATION\n";

    cout << "Name: Aditya Anand\tRoll No.:20124009\t Branch: IT\n\n\n";

    int frames = 0;

    cout<<"Enter the number of frames: ";

    cin>>frames;

    if(frames<1){

        cout<<"No frames available!";

        return 0;

    }

    cout<<"Enter the size of the reference string: ";

    int n = 0;

    cin>>n;

    cout<<"Enter the order in which pages are accessed by the CPU\n";

    vector<int> ref(n);

    for(int i=0; i<n; i++){

        cin>>ref[i];

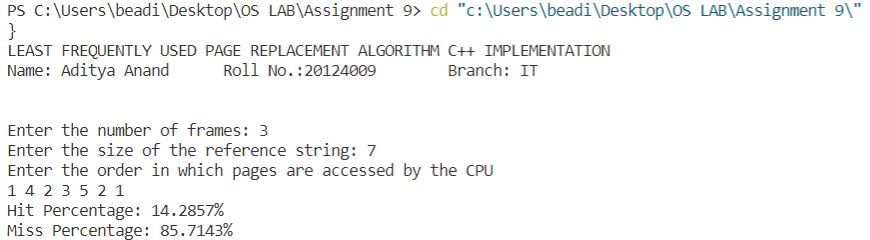
    }

    LRU(ref, frames);

    return 0;

}

RESULT:



SIMULATION OF PAGING TECHNIQUES IN MEMORY MANAGEMENT

CODE:

#include <iostream>

using namespace std;

#define MAX 50

int main(){

    cout << "SIMULATION OF PAGING TECHNIQUES C++ IMPLEMENTATION\n";

    cout << "Name: Aditya Anand\tRoll No.:20124009\t Branch: IT\n\n\n";

    int page[MAX], i, no\_of\_pgs, no\_of\_frms, pg\_sz, off, pno;

    int choice = 0;

    cout << "Enter no of pages in memory: " << endl;

    cin >> no\_of\_pgs;

    cout << "Enter page size: " << endl;

    cin >> pg\_sz;

    cout << "Enter no of frames: " << endl;

    cin >> no\_of\_frms;

    for (i = 0; i < no\_of\_frms; i++)

      page[i] = -1;

    cout << "\nEnter the page table\n";

    cout << "(Enter frame no as -1 if that page is not present in any frame)\n\n"

        << endl;

    cout << "\npageno\tframeno\n-------\t-------\n";

    for (i = 0; i < no\_of\_pgs; i++){

      cout << "\n\n"

          << i << "\t\t";

      cin >> page[i];

    }

    do{

      cout << "\n\nEnter the logical address(i.e,page no & offset): ";

      cin >> pno >> off;

      if (page[pno] == -1)

        cout << "\n\nThe required page is not available in any of frms";

      else

        cout << "Physical address (i.e, frame no and offset) : " << page[pno] << off << endl;

      cout << "\nDo you want to continue(1/0)?:";

      cin >> choice;

    }while (choice == 1);

    return 0;

}

RESULT:

