**LAB – 9**

Name: **Gandevia Keval Dharmeshbhai**

Subject: **CSA**

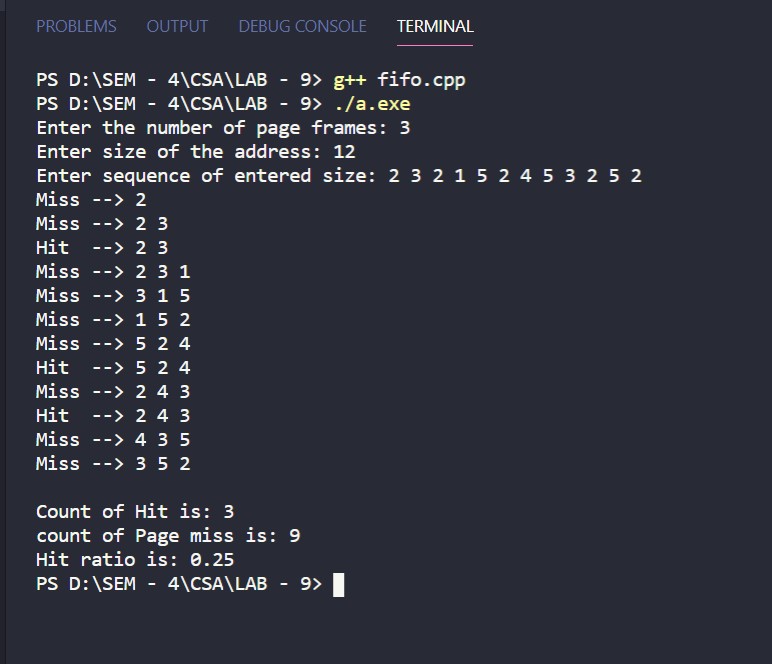
Roll No: **CE037**

ID: **19CEUEG017**

**Q.1: Implement replacement policy using FIFO.**

* **Code:**

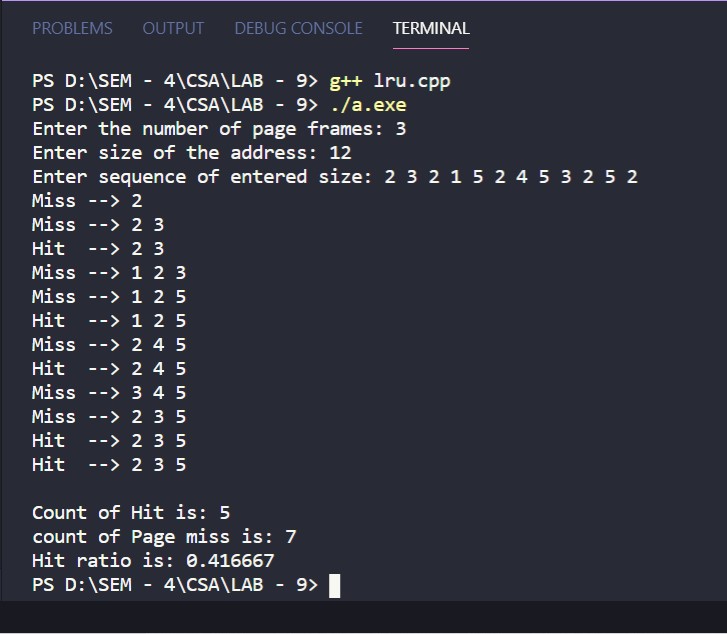
|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  // function to check page is present or not  bool isPresent(queue<int> q, int number)  {  queue<int> temp\_queue = q;  while (!temp\_queue.empty())  {  if (temp\_queue.front() == number)  {  return true;  }  temp\_queue.pop();  }  return false;  }  // display queue elements.  void showQueue(queue<int> q)  {  queue<int> temp\_queue = q;  while (!temp\_queue.empty())  {  cout << temp\_queue.front() << " ";  temp\_queue.pop();  }  cout << endl;  }  // calculate hit ratio.  void hitRatio(int page\_frame, vector<int> address)  {  int hit\_count = 0;  queue<int> q;  for (int i = 0; i < address.size(); i++)  {  // is present  if (isPresent(q, address[i]))  {  hit\_count++;  cout << "Hit --> ";  }  else  {  cout << "Miss --> ";  // replace  if (q.size() == page\_frame)  {  q.pop();  q.push(address[i]);  }  else  {  q.push(address[i]);  }  }  showQueue(q);  }  double hit\_ratio = (double)hit\_count / (double)address.size();  cout << endl  << "Count of Hit is: " << hit\_count << endl;  cout << "count of Page miss is: " << address.size() - hit\_count << endl;  cout << "Hit ratio is: " << hit\_ratio << endl;  }  int main()  {  int page\_frame, size\_of\_address, i;  cout << "Enter the number of page frames: ";  cin >> page\_frame;  cout << "Enter size of the address: ";  cin >> size\_of\_address;  vector<int> address(size\_of\_address);  cout << "Enter sequence of entered size: ";  for (i = 0; i < size\_of\_address; i++)  {  cin >> address[i];  }  hitRatio(page\_frame, address);  return 0;  } |

* **Input and output:**

**Q.2: Implement replacement policy using LRU.**

* **Code:**

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  // function to check page is present or not  bool isPresent(map<int, int> m, int number)  {  for (auto it : m)  {  if (it.first == number)  {  return true;  }  }  return false;  }  // display elements.  void showMap(map<int, int> m)  {  for (auto it : m)  {  cout << it.first << " ";  }  cout << endl;  }  // calculate hit ratio.  void hitRatio(int page\_frame, vector<int> address)  {  int hit\_count = 0;  map<int, int> m;  for (int i = 0; i < address.size(); i++)  {  // is present.  if (isPresent(m, address[i]))  {  hit\_count++;  m[address[i]] = i;  cout << "Hit --> ";  }  else  {  cout << "Miss --> ";  // replace  if (m.size() == page\_frame)  {  int min = INT\_MAX;  for (auto it : m)  {  if (it.second < min)  {  min = it.second;  }  }  for (auto it : m)  {  if (it.second == min)  {  m.erase(it.first);  m.insert({address[i], i});  break;  }  }  }  else  {  m.insert({address[i], i});  }  }  showMap(m);  }  double hit\_ratio = (double)hit\_count / (double)address.size();  cout << endl  << "Count of Hit is: " << hit\_count << endl;  cout << "count of Page miss is: " << address.size() - hit\_count << endl;  cout << "Hit ratio is: " << hit\_ratio << endl;  }  int main()  {  int page\_frame, size\_of\_address, i;  cout << "Enter the number of page frames: ";  cin >> page\_frame;  cout << "Enter size of the address: ";  cin >> size\_of\_address;  vector<int> address(size\_of\_address);  cout << "Enter sequence of entered size: ";  for (i = 0; i < size\_of\_address; i++)  {  cin >> address[i];  }  hitRatio(page\_frame, address);  return 0;  } |

* **Input and output:**

**Q.3: Implement replacement policy using OPTIMAL.**

* **Code:**

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  // function to check page is present or not  bool isPresent(vector<int> v, int number)  {  for (int i = 0; i < v.size(); i++)  {  if (v[i] == number)  {  return true;  }  }  return false;  }  // display elements.  void showVector(vector<int> v)  {  for (int i = 0; i < v.size(); i++)  {  cout << v[i] << " ";  }  cout << endl;  }  // replace elements  int replace(vector<int> v, vector<int> address, int index)  {  int j, max = index, res = 0;  for (int i = 0; i < v.size(); i++)  {  for (j = index + 1; j < address.size(); j++)  {  if (v[i] == address[j])  {  if (j > max)  {  max = j;  res = i;  }  break;  }  }  if (j == address.size())  {  res = i;  break;  }  }  return res;  }  // calculate hit ratio.  void hitRatio(int page\_frame, vector<int> address)  {  int hit\_count = 0;  // map<int,int> m;  vector<int> v;  for (int i = 0; i < address.size(); i++)  {  // is present.  if (isPresent(v, address[i]))  {  hit\_count++;  cout << "Hit --> ";  }  else  {  cout << "Miss --> ";  // replace elements  if (v.size() == page\_frame)  {  int index = replace(v, address, i);  v[index] = address[i];  }  else  {  v.push\_back(address[i]);  }  }  showVector(v);  }  double hit\_ratio = (double)hit\_count / (double)address.size();  cout << endl  << "Count of Hit is: " << hit\_count << endl;  cout << "count of Page miss is: " << address.size() - hit\_count << endl;  cout << "Hit ratio is: " << hit\_ratio << endl;  }  int main()  {  int page\_frame, size\_of\_address, i;  cout << "Enter the number of page frames: ";  cin >> page\_frame;  cout << "Enter size of the address: ";  cin >> size\_of\_address;  vector<int> address(size\_of\_address);  cout << "Enter sequence of entered size: ";  for (i = 0; i < size\_of\_address; i++)  {  cin >> address[i];  }  hitRatio(page\_frame, address);  return 0;  } |

* **Input and output:**

