

**TRƯỜNG ĐẠI HỌC SÀI GÒN**

**KHOA CÔNG NGHỆ THÔNG TIN**



**TIỂU LUẬN – HỌC PHẦN CƠ SỞ DỮ**

**LIỆU NÂNG CAO**

Họ và tên: Phạm Minh Quân

MSSV: 3120410438

Lớp môn học: DCT1201

Năm học: 2022 – 2023, Học kì: 1

Giảng viên giảng dạy: PGS. TS. Nguyễn Tuấn Đăng

***Thành phố Hồ Chí Minh - Tháng 11/2022***

**Mục lục**

[**Chương 1: Giới thiệu mục tiêu của đề tài tiểu luận** 1](#_Toc121570569)

[**Chương 2: Trình bày cơ sở lý thuyết** 2](#_Toc121570570)

[Primary index: 2](#_Toc121570571)

[Clustering index: 3](#_Toc121570572)

[Secondary index: 4](#_Toc121570573)

[**TỔNG KẾT** 6](#_Toc121570574)

[**TÀI LIỆU THAM KHẢO:** 6](#_Toc121570575)

[**Phụ lục:** 7](#_Toc121570576)

[Lưu đồ thuật toán 7](#_Toc121570577)

[Excel 7](#_Toc121570578)

[Lời giải các bài 7](#_Toc121570579)

[**Bài 1:** 8](#_Toc121570580)

[**Bài 2:** 8](#_Toc121570581)

[**Bài 3:** 9](#_Toc121570582)

[CODE: 9](#_Toc121570583)

# **Chương 1: Giới thiệu mục tiêu của đề tài tiểu luận**

Trong bài tiểu luận này, chúng ta sẽ cùng nhau tìm hiểu cấu trúc truy cập phụ trợ được gọi là chỉ mục (index). Các dữ liệu trong file được lưu trữ trong bộ nhớ phụ (đĩa) dữa trên khối lượng khổng lồ của file. Để lấy các dữ liệu cần thiết trong file, chúng cần phải đọc một vài block từ đĩa tới bộ nhớ chính trước khi lấy các block cần thiết. Chỉ mục dùng để tăng tốc độ truy cập của các bản ghi được đáp ứng các điều kiện tìm kiếm nhất định. Và cấu trúc chỉ mục sẽ giúp truy cập hiệu quả hơn (truy cập ít block hơn) dựa trên các trường lập chỉ mục được sử dụng để tạo chỉ mục. Cấu trúc của chỉ mục là các tệp bổ sung trên file với tác dụng cung cấp các đường truy cập khác đến các bản ghi mà không ảnh hưởng tới vị trí vật lí của các bản ghi chính trong file. Về cơ bản, bất kỳ trường nào của file đều có thể được sử dụng để tạo chỉ mục và nhiều chỉ mục trên các trường khác nhau cũng như các chỉ mục trên nhiều trường có thể được tạo trên cùng một file. Có thể có nhiều chỉ mục khác nhau, mỗi chỉ mục sử dụng cấu trúc dữ liệu riêng để tăng tốc độ tìm kiếm. Để tìm kiếm một hoặc nhiều bản ghi trong file dựa trên điều kiện tìm kiếm từ các trường lập chỉ mục, khi đó, chỉ mục tìm kiếm các con trỏ dẫn đến một hoặc nhiều khối đĩa trong file chứa các bản ghi cần thiết. Các chỉ mục cũng có thế được xây dựng dựa trên hàm hash hoặc cấu trúc tìm kiếm dữ liệu khác. Chúng ta sẽ mô tả các loại chỉ mục được sắp xếp theo các loại single-level khác nhau: Primary index, Clustering index, Secondary index.

+Primary index: Được chỉ định trên trường khóa sắp xếp của một ordered file được sắp xếp.

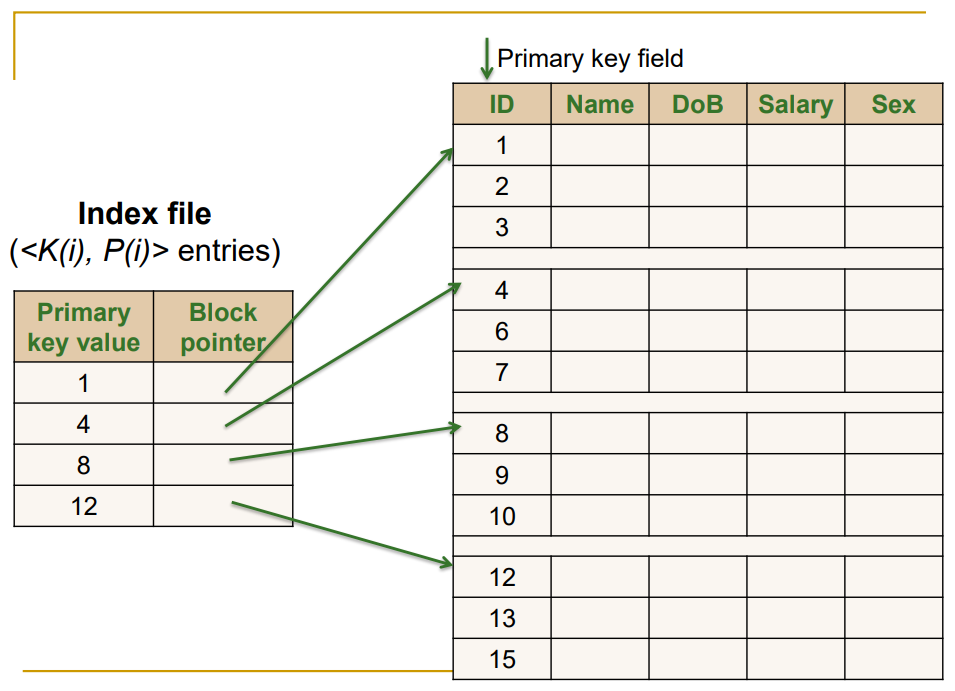
+Clustering index: Nếu trường chỉ định không có khóa của trường.

+Secondary index: Phụ thuộc vào trường chỉ định hay trường không

# **Chương 2: Trình bày cơ sở lý thuyết**

## Primary index:

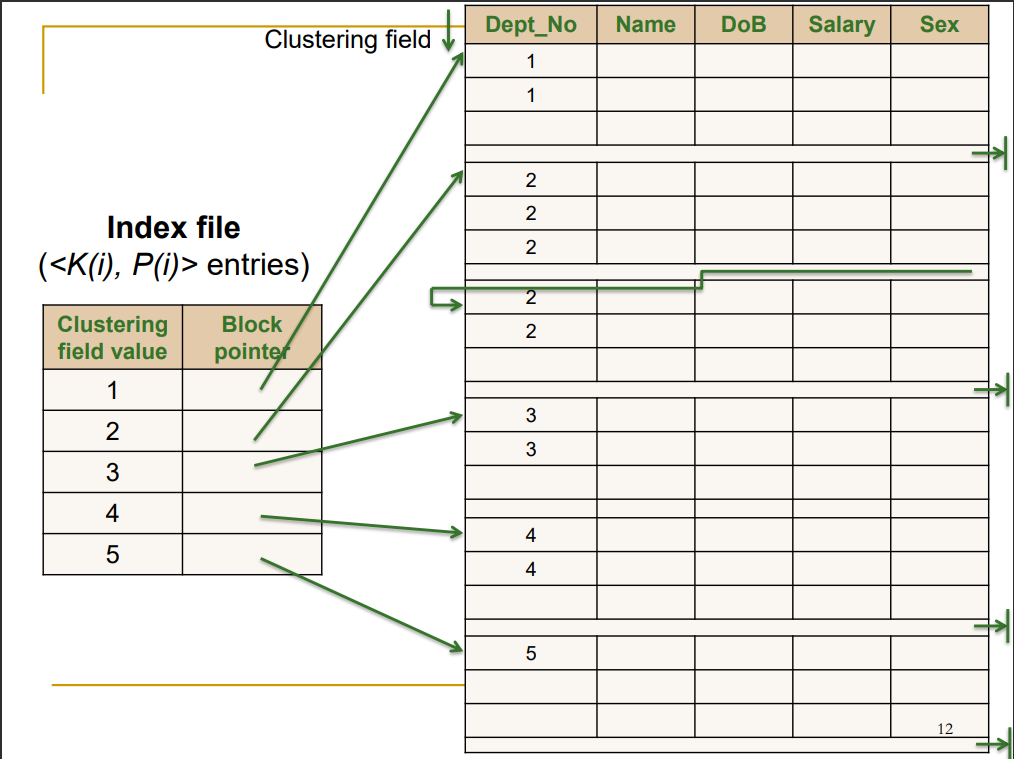
Đầu tiên, chúng ta đến với Primary index. Primary index là một file mà độ dài của nó được điều chỉnh bằng với 2 trường. Primary index hoạt động như một cấu trúc truy cập tìm kiếm một cách hiệu quả hơn và truy cập vào dữ liệu trong file. Trường đầu tiên của file có cùng dữ liệu với ordering key nên được gọi là Primary key của file, và trường thứ 2 gọi là con trỏ để chỉ tới block đĩa (block access). Trong mỗi block trong file tồn tại 1 index đầu vào ( hay còn gọi là index bản ghi). Mỗi index đầu vào có giá trị primary key của mỗi trường để ghi dữ liệu trong block và trỏ tới block đó dưới giá trị của 2 trường. Bản ghi đầu tiên trong block được gọi là Block anchor và có thể sử dụng cho bản ghi cuối của block.Và số lượng của tổng index đầu vào bằng với số lượng của block đĩa trong file. Các index này còn có nét đặc trưng như là dense hoặc sparse. Dense index có index đầu vào dành cho mỗi khóa tìm kiếm (và hàng rào cản trở của mỗi bản ghi) trong file. Còn sparse index ( hay còn gọi là non-dense index) thì ngược lại, chỉ có index đầu vào cho 1 vài dữ liệu tìm kiếm. Điều đó cho thấy sparse index có số lượng index đầu vào ít hơn so với số lượng dense index. Nên là primary index thuộc dạng sparse (non-dense) index. Và tìm kiếm nhị phân trong index file cũng đòi hỏi ít block truy cập hơn tìm kiếm nhị phân trong data file



## Clustering index:

Tiếp đến, đối với bản ghi được sắp xếp vật lí trên một trường không khóa mà nó không mang giá trị phân biệt cho mỗi trường được gọi là Clustering field và file đó được gọi Clustered file. Trong file này, chúng ta có thể tạo ra các loại index khác nhau, và các index này được gọi là clustering index. Chúng dùng để tăng tốc độ truy cập tất cả các bản ghi có giả trị trùng với trường clustering field.

Khác với primary index, clustering index đòi hỏi phải có giá trị phân biệt cho mỗi bản ghi trong ordering field của file. Và clustering index cũng là một ordered file với hai trường. Trường thứ nhất có cùng dữ liệu với clustering field và trường thứ 2 là con trỏ block đĩa. Chỉ có một index đầu vào trong clustering index cho mỗi giá trị phân biệt của clustering field, và con trỏ chỉ tới block đầu tiên trong file chứa bản ghi có giá trị phân biệt cho clustering field của nó. Đối với clustering index, thêm và xóa vẫn xảy ra lỗi, vì bản ghi dữ liệu được sắp xếp vật lí.



## Secondary index:

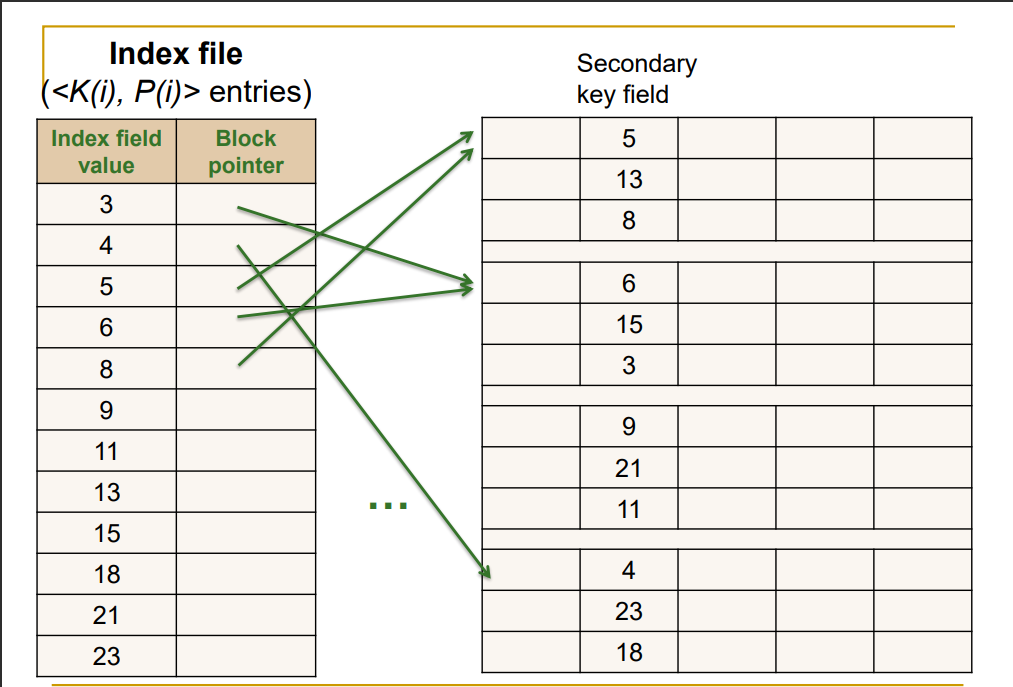
Tới chỉ mục thứ 3, secondary index cung cấp cho file một hướng truy cập dữ liệu khác khi file đã có vài quyền truy cập chính đã tồn tại. Đối với secondary index, dữ liệu trong bản ghi file có thể được sắp xếp hoặc không sắp xếp hoặc bị chia. Ngoài ra, secondary index có thể tạo ra các trường có tác dụng là khóa ứng viên và có giả trị duy nhất trong mỗi bản ghi, hoặc có trị trùng lắp trên trường không khóa. Và một lần nữa, nó lại là ordered field với 2 trường. Trường thứ nhất có cùng dữ liệu với non-ordering field của file được gọi là indexing field. Còn trường thứ 2 thì hoặc là block con trỏ hoặc là bản ghi con trỏ. Nhiều secondary index ( và hàng rào cản trở, indexing fields) có thể tạo ra một phương tiện truy cập khác cho cùng file dựa trên một số trường cụ thể. Trước tiên, chúng ta phải xác định 1 secondary index truy cập vào key (unique) field mà nó có giá trị phân biệt cho mỗi bản ghi. Và mỗi vùng đó được gọi là khóa phụ. Trong mô hình quan hệ, nó có thể tương ứng với bất kì thuộc tính unique key nào hoặc là thuộc tính khóa chính của table. Trong trường hợp này, một index đầu vào cho mỗi bản ghi trong file sẽ chứa giá trị của trường cho bản ghi và con trỏ hoặc block trong bản ghi được lưu trữ bởi chính nó. Vì vậy, secondary index thuộc loại dense.

Chúng ta cũng có thể tạo một secondary index trên 1 nonkey, 1 non-ordering field của file. Trong trường hợp đó, số lượng bản ghi trong file có thể sẽ có vài giá trị giống với indexing field. Và có 1 số lựa chọn cho việc triển khai một secondary index:

+Lựa chọn 1: là dùng các index đầu vào trùng lắp với giá trị của mỗi bản ghi. Nó có thể là dense index.

+Lựa chọn 2: là bản ghi có độ dài thay đổi theo index đầu vào, với các trường lặp lại cho con trỏ. Trong lực chọn 1 hoặc 2, tìm kiếm nhị phân trên index phải được sửa đổi thích hợp để tính số lượng index đầu vào cho mỗi giá trị của key index.

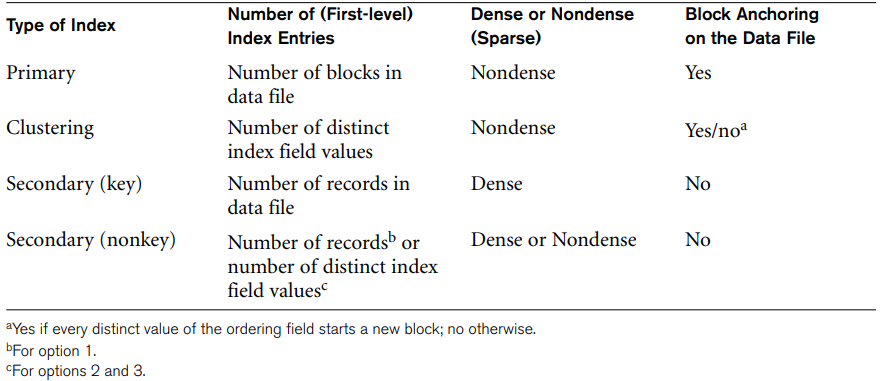
+Đến với lựa chọn 3: đây là lựa chọn được sử dụng phổ biến nhất trong cả 3. Đó là giữ cho index đầu vào được chỉnh sửa theo độ dài cố định và có 1 index đầu vào cho mỗi giá trị của index field.



Lưu ý rằng secondary index chỉ cung cấp cho thứ tự hợp lí trên bản ghi của indexing field. Nếu chúng ta truy cập vào bản ghi theo thứ tự của các index đầu vào trong secondary index, chúng ta lấy chúng theo thứ tự của indexing field. Primary và clustering index được cho là trường của chúng sử dụng cho sắp xếp thứ tự vật lí của bản ghi trong file tương tự như indexing field

# **TỔNG KẾT**

Từ 3 mục trên chúng ta có thể tóm gọn bằng 1 bảng thống kê như sau:



# **TÀI LIỆU THAM KHẢO:**

1. Tài liệu:

+Index Structures For File

+Chapter 2 Indexing Structures For File

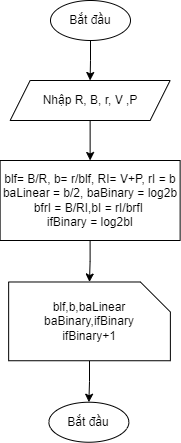
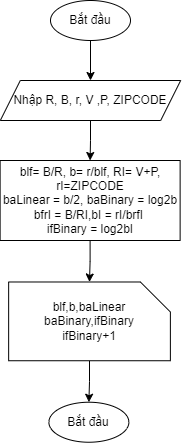
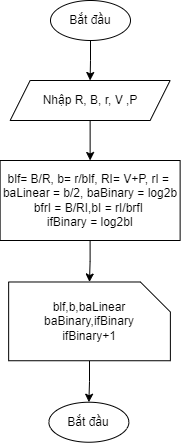
1. Website:

+ [http://cs.boisestate.edu/~jhyeh/cs410/cs410\_notes\_ch14.pdf ngày 9/12/2022](http://cs.boisestate.edu/~jhyeh/cs410/cs410_notes_ch14.pdf%20ngày%209/12/2022)

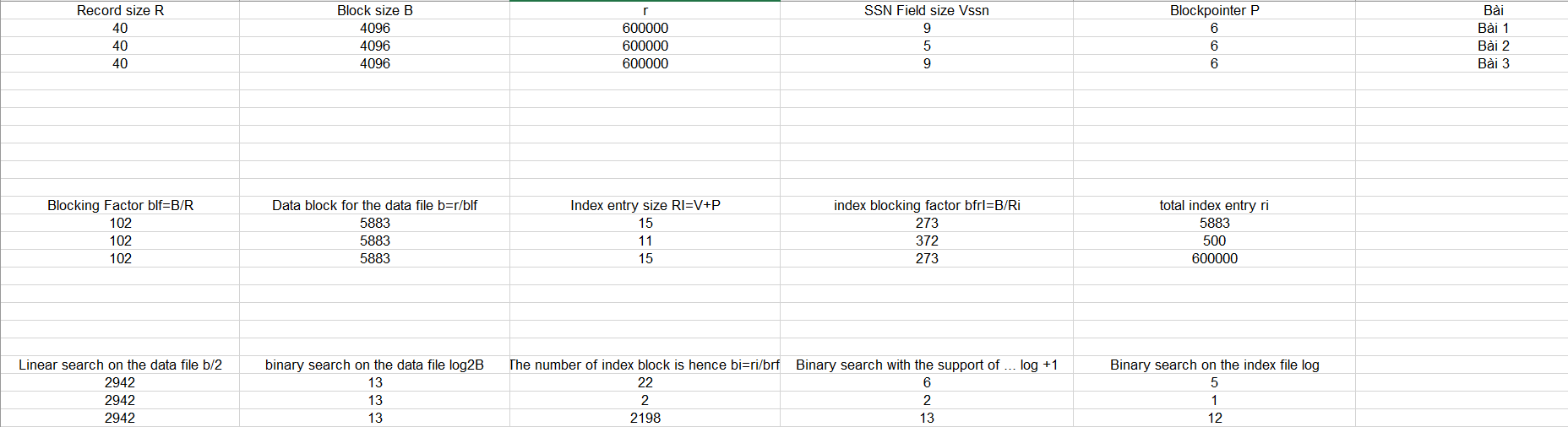
# **Phụ lục:**

## Lưu đồ thuật toán

**PRIMARY INDEX: CLUSTERING INDEX: SECONDARY INDEX:**

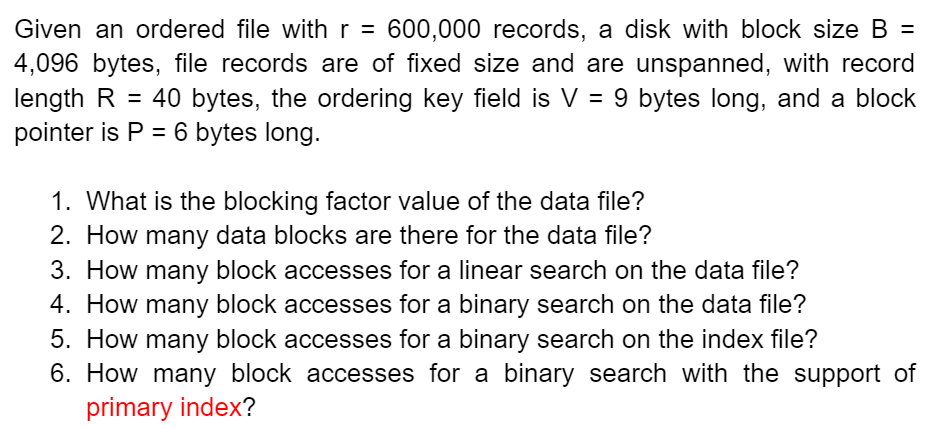
**  **

## Excel

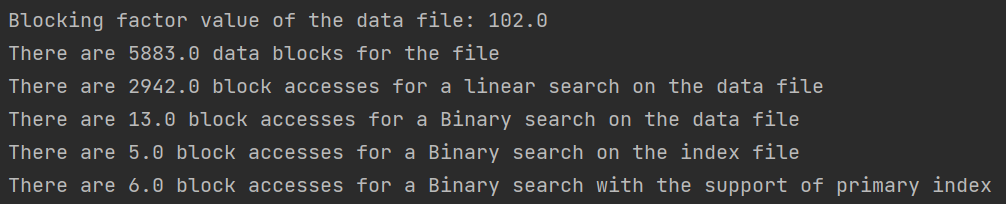


## Lời giải các bài

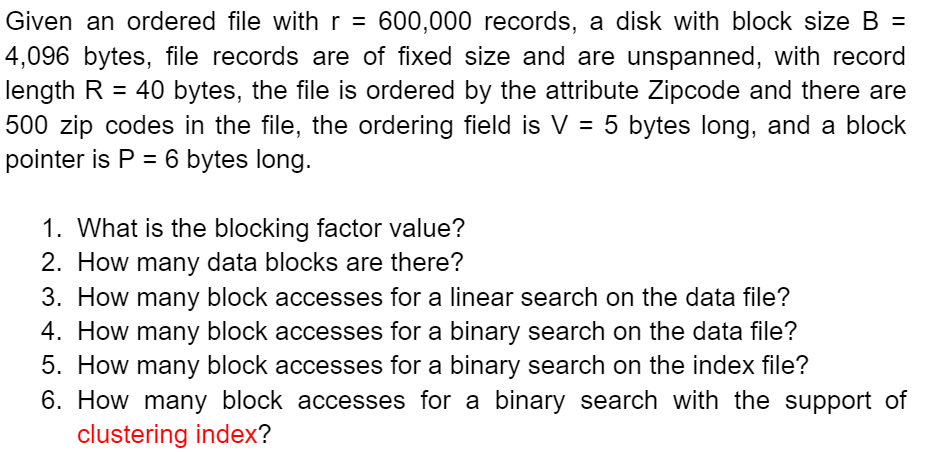
### **Bài 1:**



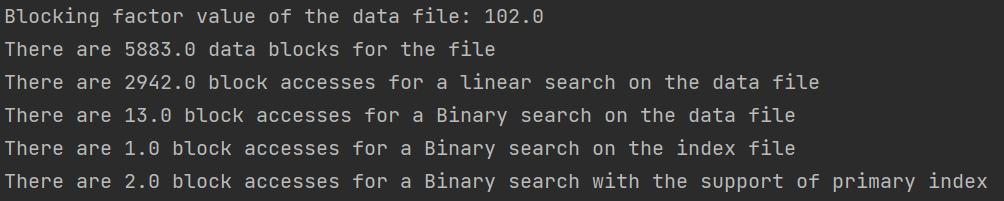
Lời giải:



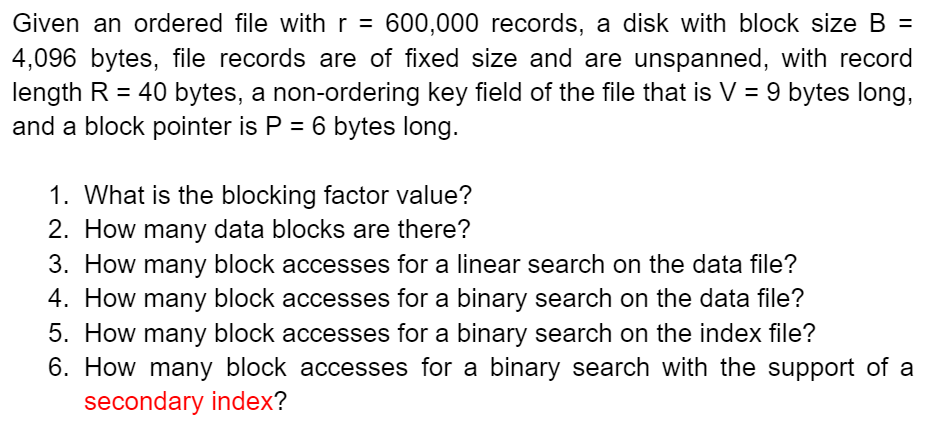
### **Bài 2:**



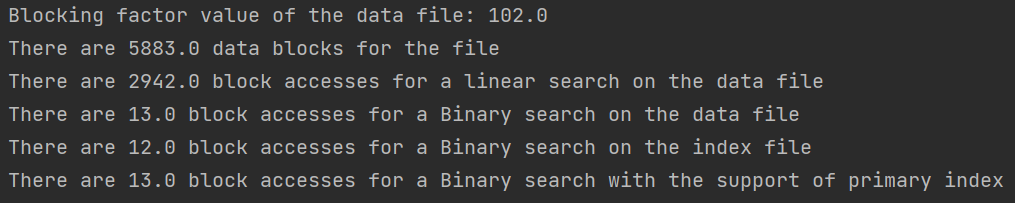
Lời giải:



### **Bài 3:**



Lời giải:



## CODE:

PRIMARY INDEX:

package CSDLNC;  
  
import java.util.Scanner;  
  
public class PRIMARYINDEX {  
 Scanner sc = new Scanner(System.*in*);  
 protected double R, B, r, V, P;  
  
 void Nhap() {  
 System.*out*.println("Nhap Record size R: ");  
 R = sc.nextInt();  
 System.*out*.println("Nhap Block size B: ");  
 B = sc.nextInt();  
 System.*out*.println("Nhap r: ");  
 r = sc.nextInt();  
 System.*out*.println("Nhap key field V: ");  
 V = sc.nextInt();  
 System.*out*.println("Nhap block pointer P: ");  
 P = sc.nextInt();  
 }  
  
 void Xuat(){  
 double blf = Math.*round*(B/R);  
 double b = Math.*round*(r/blf)+1;  
 double Ri = V + P;  
 double ri = b;  
 double baLinear = Math.*round*(b/2);  
 double baBinary = Math.*round*(Math.*log*(b)/Math.*log*(2));  
 double bfri = B/Ri;  
 double bi = ri/bfri;  
 double ifBinary = Math.*round*(Math.*log*(bi)/Math.*log*(2))+1;  
  
 System.*out*.println("Blocking factor value of the data file: " + blf);  
 System.*out*.println("There are " + b + " data blocks for the file");  
 System.*out*.println("There are " + baLinear + " block accesses for a linear search on the data file");  
 System.*out*.println("There are " + baBinary + " block accesses for a Binary search on the data file");  
 System.*out*.println("There are " + ifBinary + " block accesses for a Binary search on the index file");  
 System.*out*.println("There are " + (ifBinary+1) + " block accesses for a Binary search with the support of primary index");  
 }  
}

CLUSTERING INDEX:

package CSDLNC;  
  
import java.util.Scanner;  
  
public class CLUSTERINGINDEX {  
 Scanner sc = new Scanner(System.*in*);  
 protected int R, B, r, V, P,ZIPCODE;  
  
 void Nhap() {  
 System.*out*.println("Nhap Record size R: ");  
 R = sc.nextInt();  
 System.*out*.println("Nhap Block size B: ");  
 B = sc.nextInt();  
 System.*out*.println("Nhap r: ");  
 r = sc.nextInt();  
 System.*out*.println("Nhap key field V: ");  
 V = sc.nextInt();  
 System.*out*.println("Nhap block pointer P: ");  
 P = sc.nextInt();  
 System.*out*.println("Nhap Zipcode: ");  
 ZIPCODE = sc.nextInt();  
 }  
  
 void Xuat(){  
 double blf = Math.*round*(B/R);  
 double b = Math.*round*(r/blf)+1;  
 double Ri = V + P;  
 double ri = ZIPCODE;  
 double baLinear = Math.*round*(b/2);  
 double baBinary = Math.*round*(Math.*log*(b)/Math.*log*(2));  
 double bfri = B/Ri;  
 double bi = ri/bfri;  
 double ifBinary = Math.*round*(Math.*log*(bi)/Math.*log*(2))+1;  
  
 System.*out*.println("Blocking factor value of the data file: " + blf);  
 System.*out*.println("There are " + b + " data blocks for the file");  
 System.*out*.println("There are " + baLinear + " block accesses for a linear search on the data file");  
 System.*out*.println("There are " + baBinary + " block accesses for a Binary search on the data file");  
 System.*out*.println("There are " + ifBinary + " block accesses for a Binary search on the index file");  
 System.*out*.println("There are " + (ifBinary+1) + " block accesses for a Binary search with the support of primary index");  
 }  
}

SECONDARY INDEX:

package CSDLNC;  
  
import java.util.Scanner;  
  
public class SECONDARYINDEX {  
 Scanner sc = new Scanner(System.*in*);  
 protected int R, B, r, V, P;  
  
 void Nhap() {  
 System.*out*.println("Nhap Record size R: ");  
 R = sc.nextInt();  
 System.*out*.println("Nhap Block size B: ");  
 B = sc.nextInt();  
 System.*out*.println("Nhap r: ");  
 r = sc.nextInt();  
 System.*out*.println("Nhap key field V: ");  
 V = sc.nextInt();  
 System.*out*.println("Nhap block pointer P: ");  
 P = sc.nextInt();  
 }  
  
 void Xuat(){  
 double blf = Math.*round*(B/R);  
 double b = Math.*round*(r/blf)+1;  
 double Ri = V + P;  
 double ri = r;  
 double baLinear = Math.*round*(b/2);  
 double baBinary = Math.*round*(Math.*log*(b)/Math.*log*(2));  
 double bfri = B/Ri;  
 double bi = ri/bfri;  
 double ifBinary = Math.*round*(Math.*log*(bi)/Math.*log*(2))+1;  
  
 System.*out*.println("Blocking factor value of the data file: " + blf);  
 System.*out*.println("There are " + b + " data blocks for the file");  
 System.*out*.println("There are " + baLinear + " block accesses for a linear search on the data file");  
 System.*out*.println("There are " + baBinary + " block accesses for a Binary search on the data file");  
 System.*out*.println("There are " + ifBinary + " block accesses for a Binary search on the index file");  
 System.*out*.println("There are " + (ifBinary+1) + " block accesses for a Binary search with the support of primary index");  
 }  
}

MAIN:

package CSDLNC;  
  
import java.util.Scanner;  
  
public class Xuly {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 int choice;  
 do {  
 System.*out*.println("--------------");  
 System.*out*.println("1. Bai tap 1");  
 System.*out*.println("2. Bai tap 2");  
 System.*out*.println("3. Bai tap 3");  
 System.*out*.println("4. Thoat");  
 System.*out*.println("---------------");  
 System.*out*.println("Nhap lua chon: ");  
 choice = sc.nextInt();  
 switch (choice) {  
 case 1: {  
 PRIMARYINDEX P = new PRIMARYINDEX();  
 P.Nhap();  
 P.Xuat();  
 break;  
 }  
 case 2: {  
 CLUSTERINGINDEX C = new CLUSTERINGINDEX();  
 C.Nhap();  
 C.Xuat();  
 break;  
 }  
 case 3: {  
 SECONDARYINDEX S = new SECONDARYINDEX();  
 S.Nhap();  
 S.Xuat();  
 break;  
 }  
 case 4: {  
 System.*out*.println("Dang thoat chuong trinh");  
 System.*exit*(0);  
 }  
 }  
 }while (choice>=1 || choice<=4);  
 }  
}

# 