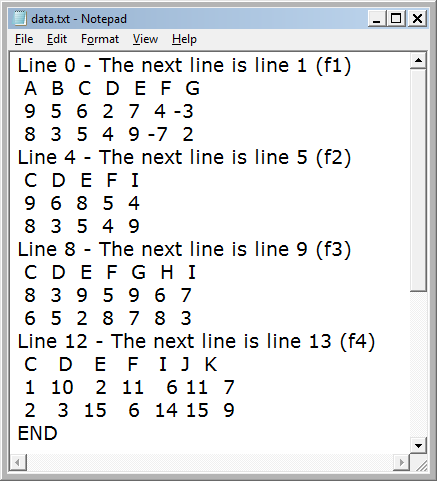
**Giải thích các File được cung cấp khi làm**

**bài thi thực hành (PE) môn CSD201**

Dữ liệu đầu vào và đầu ra của bài thi là file văn bản. Tập tin chứa dữ liệu đầu vào thường có dạng:



Do vậy, thư viện được cung cấp chủ yếu là các tác vụ liên quan đến truy xuất file văn bản theo cơ chế từng dòng phân cách các từ/ số bằng ký tự khoảng trống (các dòng được đánh số từ 0 trở đi- xem file mẫu ở trên). Thư viện này chủ yếu chứa các công cụ cho việc setup các cấu trúc dữ liệu ban đầu (list/ tree/ graph- đọc file dữ liệu đưa vào cấu trúc dữ liệu) và các công cụ cho việc truy xuất file khi đánh giá bài làm. SINH VIÊN CÓ THỂ KHÔNG CẦN ĐỌC THƯ VIỆN NÀY NHƯNG NẾU ĐỌC THÌ SINH VIÊN HIỂU RÕ HƠN CODE ĐÃ ĐƯỢC CHO ĐỂ TỰ TIN LÀM BÀI THI VÀ CŨNG KHÔNG NÊN SỬA CODE THƯ VIỆN NÀY.

Như vậy, nếu một danh sách có 7 objects, mỗi object có 3 fields, các field1 được để thành 1 dòng (C D E F I J K), các field2 được để thành 1 dòng (1 10 2 11 6 11 7), các field3 được để thành 1 dòng (2 3 15 6 14 15 9) và trị đầu của 3 dòng, thí dụ (C, 1, 2), là một object. Đây chính là cấu trúc lưu data của các object lên file văn bản của mô hình này.

**GIỚI THIỆU THƯ VIỆN ĐÃ CHO:**

Tên lớp: **class Lib**

**Các methods:**

//Xuất nội dung một file văn bản ra màn hình, tham số: tên file

static void viewFile(String fname) throws Exception

//Lấy dòng thứ k trong file văn bản, tham số: tên file

static public String readLineToStr(String fname, int k)

//Lấy dòng thứ k trong file văn bản, tham số: tên file, trả trị: mảng các chuỗi con

Trong file: “C D E F I J K” 🡪 Trả về: {“C”, “D”, “E”, “F”, “I”, “J”, “K” }

static public String [] readLineToStrArray(String fname, int k)

//Lấy dòng thứ k trong file văn bản, tham số: tên file, trả trị: mảng số nguyên

Trong file: “1 10 2 11 6 11 7” 🡪 Trả về: {1, 10, 2, 11, 6, 11, 7}

static public int [] readLineToIntArray(String fname, int k)

//Lấy dòng thứ k trong file văn bản, tham số: tên file, trả trị: mảng số thực

Trong file: “1 10 2 11 6 11 7” 🡪 Trả về: {1.0, 10.0, 2.0, 11.0, 6.0, 11.0, 7.0}

static public double [] readLineToDoubleArray(String fname, int k)

//Xuất 1 chuỗi ra màn hình

static public void dispStr(String s)

//Xuất 1 mảng các chuỗi ra màn hình

static public void dispStrArray(String a[])

//Xuất các phần tử từ chỉ số k đến chỉ số h của mảng các chuỗi ra màn hình

static public void dispStrArray(String a[], int k, int h)

//Xuất mảng số int ra màn hình

static public void dispIntArray(int a[])

//Xuất các phần tử từ chỉ số k đến chỉ số h của mảng số int ra màn hình

static public void dispIntArray(int a[], int k, int h)

// Ghi chuỗi s lên file văn bản với tên file là out\_fname

static public void dispStr(String s, String out\_fname)

// Ghi chuỗi s lên file văn bản lên file đã mở f

static public void dispStr(String s, RandomAccessFile f)

//Ghi các phần tử từ chỉ số k đến chỉ số h của mảng chuỗi lên file có tên out\_fname

static public void dispStrArray(String a[], int k, int h, String out\_fname)

//Ghi các phần tử từ chỉ số k đến chỉ số h của mảng chuỗi lên file đã mở f

static public void dispStrArray(String a[], int k, int h, RandomAccessFile f)

//Ghi toàn bộ mảng chuỗi a[] lên file có tên out\_fname

static public void dispStrArray(String a[], String out\_fname)

//Ghi toàn bộ mảng chuỗi a[] lên file đã mở f

static public void dispStrArray(String a[], RandomAccessFile f)

//Ghi các phần tử từ chỉ số k đến chỉ số h của ảng int a[] lên file có tên out\_fname

static public void dispIntArray(int a[], int k, int h, String out\_fname)

//Ghi các phần tử từ chỉ số k đến chỉ số h của ảng int a[] lên file đã mở f

static public void dispIntArray(int a[], int k, int h, RandomAccessFile f)

//Ghi toàn bộ mảng int a[] lên file có tên out\_fname

static public void dispIntArray(int a[], String out\_fname)

//Ghi toàn bộ mảng int a[] lên file đã mở f

static public void dispIntArray(int a[], RandomAccessFile f)

**DẠNG CHUNG CỦA ĐỀ THI PE**

Ba chủ đề sẽ thi thực hành được quy định trong Syllabus của môn CSD201: Linked Lists, Binary Search Trees và Graphs. Dạng file thi PE nhìn chung như sau: Sinh viên được cung cấp các project trong đó có các methods đã hoàn tất, sinh viên cần biết để có thể dùng tới, một số methods chưa hoàn thành trong một class nào đó cần phải được hoàn thành. Dựa trên yêu cầu của đề thi, sinh viên cần điền code vào các methods chưa hoàn thành này để chương trình chạy được và kết quả của chương trình phải giống với đáp án. Input và output của mỗi bài đều là file văn bản (xem hàm main() để biết tên các file này)

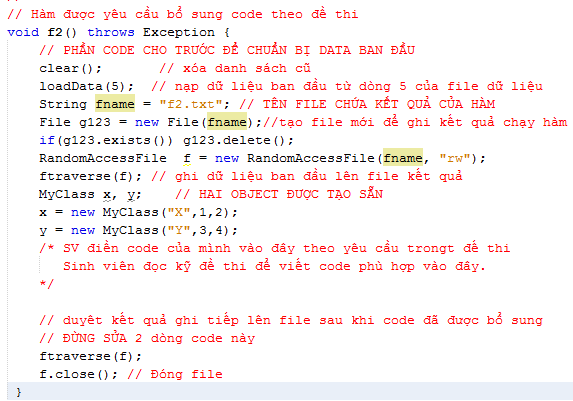
Khi làm bài, sinh viên có thể tự tạo thêm hàm mới để dùng và đừng lo ảnh hưởng lên chương trình.

**Hình thức chung của bài thi về Linked Lists:**

**Cấu trúc file của đề thi:**

|  |  |
| --- | --- |
|  | **Thing.java**: Class cho một vật gì đó  **Node.java**: Class cho một nút trong linked list, thường là Singly LL, chứa data là **Thing**.  **MyList.java**: Class cho một linked list được quản lý bằng head và tail. Mỗi phần tử trong list là một **Node**. Thông thường, trong lớp này có một số methods đã có code hoàn chỉnh (SV không nên sửa) và một số method chưa hoàn tất và đòi hỏi SV phải điền thêm code vào những vùng được quy định dựa trên yêu cầu của đề bài.  **Lib.java**: Lớp công cụ đã được giới thiệu ở trước.  **Main.java**: Class có hành vi main() để hình thành chương trình.  **Data.txt**: Tập tin chứa dữ liệu đầu vào của chương trình. |
| Sau khi điền code xong, SV có thể chạy chương trình. Đẩu ra của chương trình là những tập tin đã được quy định trong hàm main(). SV có thể nhìn thấy kêt quả của chương trình trên màn hình hoặc mở các tập tin kết quả (file văn bản) để xem kết quả đã làm của mình. | |

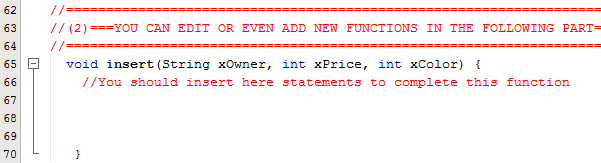
***Dạng method mà sinh viên có thể gặp như sau:***

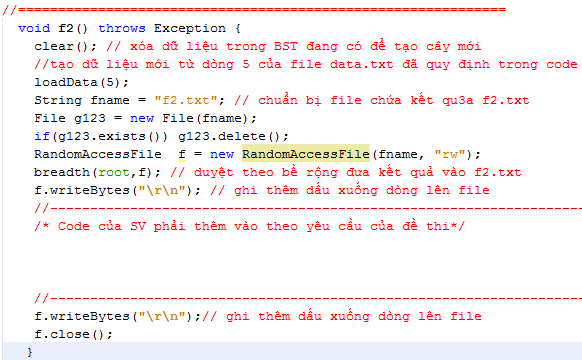


**Hình thức chung cho mỗi yêu cầu của bài thi về BSTs:**

|  |  |
| --- | --- |
|  | **Thing.java**: Class cho một vật gì đó  **Node.java**: Class cho một nút trong BST, chứa data là **Thing**.  **BSTree.java**: Class cho một cây BST các nodes được quản lý bằng root. Thông thường, trong lớp này có một số methods đã có code hoàn chỉnh (SV không nên sửa) và một số method chưa hoàn tất và đòi hỏi SV phải điền thêm code vào những vùng được quy định dựa trên yêu cầu của đề bài.  **Main.java**: Class có hành vi main() để hình thành chương trình.  **Queue,java**: Class cho hàng đợi, công cụ duyệt cây theo chiều rộng.  **Lib.java**: Lớp công cụ đã được giới thiệu ở trước.  **Data.txt**: Tập tin chứa dữ liệu đầu vào của chương trình. |
| Sau khi điền code xong, SV có thể chạy chương trình. Đẩu ra của chương trình là những tập tin đã được quy định trong hàm main(). SV có thể nhìn thấy kêt quả của chương trình trên màn hình hoặc mở các tập tin kết quả (file văn bản) để xem kết quả đã làm của mình. | |

***Dạng method mà sinh viên có thể gặp như sau:***

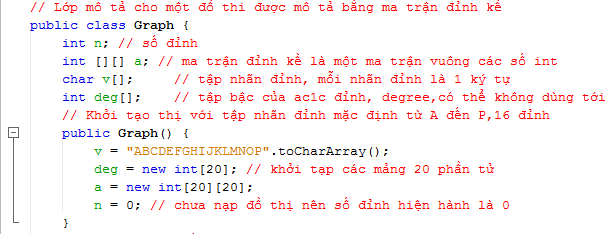




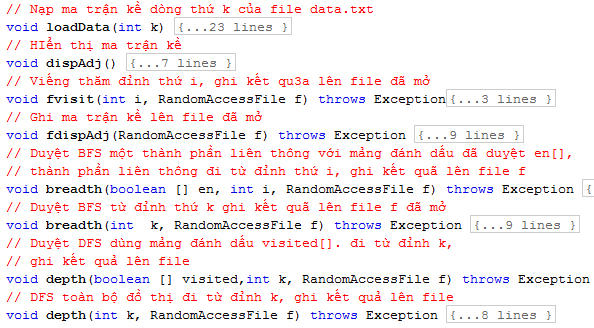
**Hình thức chung cho mỗi yêu cầu của bài thi về Graphs:**

|  |  |
| --- | --- |
|  | **Graph.java**: Class cho một đồ thị THƯỜNG được biểu diễn bằng ma trận đỉnh kề (adjacency matrix). Trong lớp này có một số methods đã có code hoàn chỉnh (SV không nên sửa) và một số method chưa hoàn tất và đòi hỏi SV phải điền thêm code vào những vùng được quy định dựa trên yêu cầu của đề bài.  **Main.java**: Class có hành vi main() để hình thành chương trình.  **Queue,java, Stack.java**: Class cho queue/stack. Các công cụ cho các thao tác duyệt đồ thi.  **Lib.java**: Lớp công cụ đã được giới thiệu ở trước.  **Data.txt**: Tập tin chứa dữ liệu đầu vào của chương trình. |
| Sau khi điền code xong, SV có thể chạy chương trình. Đẩu ra của chương trình là những tập tin đã được quy định trong hàm main(). SV có thể nhìn thấy kêt quả của chương trình trên màn hình hoặc mở các tập tin kết quả (file văn bản) để xem kết quả đã làm của mình. | |

Dạng method mà sinh viên có thể gặp như sau:



Các methods đã hoàn tất, sinh viên có thể dùng:



**Dạng method chưa hoàn thành mà sinh viên có thể gặp như sau:**

