Đại Học Quốc Gia TP.HCM
Trường Đại học Bách Khoa
Khoa KH&KT Máy Tính

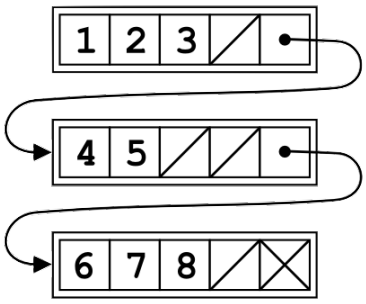
Vietnam National University – HCMC
Hochiminh University of Technology
Faculty of Computer Science and Engineering

CẤU TRÚC DỮ LIỆU & GIẢI THUẬT
ASSIGNMENT 01 - UNROLLED LINKED LIST

1 Giới thiệu

Trong bài tập lớn này, sinh viên sẽ sửa đổi danh sách liên kết đã học thành một cấu trúc dữ liệu mới là danh sách liên kết mở (unrolled linked list). Cấu trúc dữ liệu này tương tự như danh sách liên kết thông thường ngoại trừ việc có thể có nhiều hơn một phần tử được lưu tại mỗi node. Sinh viên có thể tham khảo thêm về unrolled linked list từ các tài liệu khác, tuy nhiên có thể mô tả sẽ khác ít nhiều so với assignment này.

Giống như mảng (array) và danh sách liên kết, unrolled linked list cũng là cấu trúc dữ liệu tuyến tính. Thay vì lưu một phần tử dữ liệu tại mỗi node, danh sách này lưu một mảng các phần tử tại một node. Nhờ cách này, danh sách tận dụng được lợi thế từ cả hai cấu trúc dữ liệu mảng và danh sách liên kết vì nó giảm được overhead của danh sách liên kết khi lưu nhiều phần tử trong một node và có thể thêm, xoá dễ dàng. Một unrolled linked list dcó thể được mô tả như hình bên dưới:



Hình 1: Unrolled linked list với tối đa 4 phần tử mỗi node.

Mỗi node trong danh sách chứa một array với số lượng phần tử tối đa là **maxElements**. Vị trí của phần tử trong danh sách có thể được xác định bằng tham khảo hoặc vị trí trong array. Có 2 thao tác cơ bản có thể được thực hiện trên danh sách là phép chèn (insertion) và phép xoá (deletion)

- **Chèn**: để chèn một phần tử vào vị trí cụ thể, chúng ta tìm node mà phần tử cần được đặt vào và chèn phần tử đó vào array, đồng thời tăng số phần tử trong node ($numElements$). Nếu array của node trước đó đã đầy, chúng ta sẽ tạo ra một node mới ngay sau node hiện tại và chuyển phân nữa số phần tử node hiện tại sang node mới. Thao tác này sẽ làm cho kích thước danh sách tăng lên 1.
- **Xoá**: để xoá một phần tử tại một vị trí cụ thể nào đó, chúng ta tìm node chứa phần tử đó và xoá nó ra khỏi array, giảm $numElements$. Nếu số phần tử trong array của node nhỏ hơn $ceil(maxElements/2)$, chúng ta lấy 1 phần tử từ node láng giềng coho vào node hiện tại. Nếu node láng giềng chỉ có $ceil(maxElements/2)$ phần tử thì chúng ta sẽ nối 2 node. Nếu việc hợp nhất xảy ra thì số node trong danh sách giảm 1.

Chú ý rằng có nhiều cách khác nhau để hiện thực các chức năng chèn và xoá mô tả ở trên. Tuy nhiên, trật tự của các phần tử là như nhau trong danh sách kết quả. Thêm vào đó, điều kiện về số phần tử tối thiểu trong mỗi node luôn phải thoả trừ trường hợp duy nhất là danh sách chỉ có 1 node.

Những lợi thế của Unrolled linked list:

- Nhờ cơ chế cache, danh sách có thể thực hiện duyệt tuần tự rất nhanh. Thời gian đánh chỉ mục giảm tuyến tính theo hệ số $maxElements$.

- Đòi hỏi ít không gian lưu trữ hơn.
- Thực hiện các thao tác nhanh hơn danh sách lien kết thông thường.

2 Hiện thực

Hiện thực assignment được tổ chức tập trung ở hàm **main()** và hai class **Node**, **UnrolledLinkedList**, được chia ra thành 5 file: **Node.h**, **Node.cpp**, **UnrolledLinkedList.h**, **UnrolledLinkedList.cpp**, và **main.cpp**. Chi tiết như sau:

2.1 Node.h và Node.cpp

Những file này chứa định nghĩa và hiện thực của cấu trúc dữ liệu **Node**

```
class Node {
private:
  int maxElements;
public:
  int numElements; // number of elements in this node, up to maxElements
  int* elements; // an array of numElements elements,
  Node *next; // reference to the next node in the list
  Node *prev; // reference to the previous node in the list

  Node(int capacity = 5);
  ~Node();
  int getHalfNodeSize();
  bool isUnderHalfFull();
  bool isFull();
  bool isOverflow();
  bool isEmpty();
  void add(int val); // append val element to the end of the node
  void insertAt(int pos, int val); // insert val to position pos of the node
  void removeAt(int pos); // remove the position pos from the node
  void reverse(); // reverse the content of the node
  void print(); // print the content of the node
  void printDetail(); // print the detail content of the node
};
```

2.2 UnrolledLinkedList.h

File này chứa khai báo của UnrolledLinkedList và định nghĩa các phương thức trong class. Prototype của các phương thức dựa trên hiện thực Java của danh sách liên kết [3].

```
class UnrolledLinkedList {
private:
  Node* head;
  Node* tail;
  int size;
  int numOfNodes;
  int nodeSize;
public:

  // Your tasks: complete the implementation of the below methods in UnrolledLinkedList.h
  void add(int val);
  bool contains(int val);
  int getAt(int pos);
  void setAt(int pos, int val);
  int firstIndexOf(int val);
  int lastIndexOf(int val);
  void insertAt(int pos, int val);
  void deleteAt(int pos);
  void reverse();
  int* toArray();
```

2.3 UnrolledLinkedList.cpp

File này chứa các protoptype cần thiết để sinh viên hiện thực assignment. Chi tiết các phương thức được mô tả trong phần sau.

2.4 main.cpp

File này chứa hàm main để kiểm tra hiện thực cấu trúc dữ liệu Unrolled Linked List. Hàm sẽ đọc các command từ file input.txt để tạo đối tượng UnrolledLinkedList và gọi các phương thức của class. Chi tiết như sau:

No	Command	Parameters	Activate
1	u	val (int)	new UnrolledLinkedList(val)
2	a	val (int)	add(val)
3	c	val (int)	contains(val)
4	d	pos (int)	deleteAt(pos)
5	f	val (int)	firstIndexOf(val)
6	g	pos (int)	getAt(pos)
7	i	pos (int), val (int)	insertAt(pos, val)
8	l	val (int)	lastIndexOf(val)
9	p		print()
10	r		reverse()
11	s	pos (int), val (int)	setAt(pos, val)
12	t		toArray()
13[1]	w		printDetail()

Dưới đây là một ví dụ của file input:

```
u 4
a 5
a 7
i 0 1
p
```

2.5 Yêu cầu

Yêu cầu sinh viên trong assignment là hiện thực unrolled linked list đã mô tả. Cụ thể sinh viên cần thiện thực các phương thức được cung cấp trong **UnrolledLinkedList.cpp**. Khung code được cung cấp sẵn giúp quá trình hiện thực rõ ràng hơn.

Sinh viên không được sử dụng các thư viện nào khác ngoài các thư viện đã được dùng trong code mẫu.

3 Quy định

3.1 Đánh giá

Output từ chương trình sẽ được so trùng với output kì vọng. Một testcase là đạt nếu toàn bộ output trùng khớp với output kì vọng, và không bị vi phạm ràng buộc thời gian chạy. Từng testcase sẽ được chạy và số lượng testcase đạt sẽ tương ứng với số điểm của assignment.

3.2 Nộp bài

Sinh viên sẽ nộp bài qua hệ thống và sẽ được hướng dẫn cụ thể qua thông báo trên Sakai. Chú ý, sinh viên chỉ submit một file mã nguồn UnrolledLinkedList.cpp và hệ thống sẽ build code trên Linux. Do đó sinh viên không sử dụng các hiện thực đặc biệt nào phụ thuộc hệ điều hành hay trình biên dịch.

[1] Chú ý: printDetail() chỉ được sử dụng cho mục đích debug. Hàm này sẽ không được dùng khi chấm điểm. Hàm được sử dụng để kiểm tra là print().

Bảng 1: Mô tả các yêu cầu

No	Method	Description
1	void add(int val)	Thêm một phần tử mới vào cuối danh sách Parameters: - val: phần tử thêm vào danh sách.
2	bool contains(int val)	Trả về **true** nếu danh sách chứa phần tử val. Parameters: - val: phần tử cần kiểm tra sự xuất hiện trong danh sách.
3	int getAt(int pos)	Trả về phần tử tại một vị trí xác định trong danh sách. Parameters: - pos: index của phần tử cần truy xuất Throws: - IndexOutOfBoundsException: Nếu index của pos không đúng (pos < 0 || pos >= size)
4	void setAt(int pos, int val)	Thay thế phần tử tại vị trí xác định trong danh sách. Parameters: - pos: vị trí của phần tử cần thay thế - val: giá trị của phần tử cần thay thế Throws: - IndexOutOfBoundsException : Nếu index của pos không đúng (pos < 0 || pos >= size)
5	int firstIndexOf(int val)	Trả về index của vị trí xuất hiện đầu tiên của phần tử val trong danh sách, trả về -1 nếu danh sách không chứa phần tử đó. Parameters: - val: phần tử cần tìm kiếm
6	int lastIndexOf(int val)	Trả vể index của vị trí xuất hiện cuối cùng của phần tử cần tìm trong danh sách. Parameters: - val: phần tử cần tìm kiếm
7	void insertAt(int pos, int val)	Chèn một phần tử vào vị trí xác định trong danh sách. Parameters: - pos: vị trí cần chèn phần tử - val: giá trị phần tử cần chèn Throws: - IndexOutOfBoundsException : Nếu index của pos không đúng (pos < 0 || pos > size)
8	void deleteAt(int pos)	Xoá phần tử tại 1 vị trí cụ thể trong danh sách. Parameters: - pos: index của phần tử cần xoá Throws: - IndexOutOfBoundsException : Nếu index của pos không đúng (pos < 0 || pos >= size)
9	void reverse()	Đảo ngược danh sách.
10	int* toArray()	Trả về con trỏ đến một array chứa tất cả các phần tử của danh sách theo đúng thứ tự xuất hiện trong danh sách hiện tại.

Sinh viên phải kiểm tra chương trình của mình trên Linux trước khi nộp để chắc chắn code chạy được. Hạn chót của assignment 1 là **04-11-2018**. Hệ thống chấm sẽ được mở trước ngày 04-11-2018.

3.3 Xử lý gian lận

Bài tập lớn phải được sinh viên TỰ LÀM. Sinh viên sẽ bị coi là gian lận nếu:

- Có sự giống nhau bất thường giữa mã nguồn của các bài nộp. Trong trường hợp này, tất cả các bài nộp đều bị coi là gian lận. Do vậy sinh viên phải bảo vệ mã nguồn bài tập lớn của mình. Các bài làm của các sinh viên ở các học kỳ trước cũng sẽ được dùng để kiểm tra gian lận.
- Sinh viên không hiểu mã nguồn do chính mình viết, trừ những phần mã được cung cấp sẵn trong chương trình khởi tạo. Sinh viên có thể tham khảo từ bất kỳ nguồn tài liệu nào, tuy nhiên phải đảm bảo rằng mình hiểu rõ ý nghĩa của tất cả những dòng lệnh mà mình viết. Trong trường hợp không hiểu rõ mã nguồn của nơi mình tham khảo, sinh viên được đặc biệt cảnh báo là KHÔNG ĐƯỢC sử dụng mã nguồn này; thay vào đó nên sử dụng những gì đã được học để viết chương trình.

Trong trường hợp bị kết luận là gian lận, sinh viên sẽ bị **điểm 0 cho toàn bộ môn học** (không chỉ bài tập lớn). **Không có NGOẠI LỆ nào được chấp nhận.**

Sau mỗi bài tập lớn được nộp, sẽ có một số sinh viên được gọi phỏng vấn ngẫu nhiên để chứng minh rằng bài tập lớn vừa được nộp là do chính mình làm.

4 References

1 Shao, Z.; Reppy, J. H.; Appel, A. W. (1994), "Unrolling lists", Conference record of the 1994 ACM Conference on Lisp and Functional Programming: 185–191, doi:10.1145/182409.182453, ISBN 0897916433

2 Wikipedia "Unrolled linked list"accessed 30 September 2018, Online website: https://en.wikipedia.org/wiki/Unrolled_linked_list

3 Java Implematation "Class LinkedList", accessed 30 September 2018, Online website: https://docs.oracle.com/javase/7/docs/api/java/util/LinkedList.html

4 Julie Zelenski "How assignments are graded", accessed 30 September 2018, Online website: https://web.stanford.edu/class/cs107/advice/assigngrade.html