作业一：双链表解决约瑟夫问题

一、算法思想

以双向链表存储节点信息（标号），遍历链表，每次报数到三，删除节点，打印出局信息，并重新开始报数，当遍历到尾节点之后，重新遍历

二、算法实现

编程环境：JAVA1.8

Node.java中定义节点类型，及一些基本操作

public class Node {

private int num;

private Node next;

private Node prior;

public Node(int num) {

this.num = num;

}

public int getNum() {

return num;

}

public void setNum(int num) {

this.num = num;

}

public Node getNext() {

return next;

}

public void setNext(Node next) {

this.next = next;

}

public Node getPrior() {

return prior;

}

public void setPrior(Node prior) {

this.prior = prior;

}

/\*\*

\* 在队尾增加节点node

\* @param curr 现在指向的节点

\* @param node 增加的节点

\*/

public static void increase(Node curr,Node node){

if (curr == null) {

System.out.println("给定节点为空");

return;

}

if (node == null) {

System.out.println("给定节点为空");

return;

}

while(curr.next!=null){//循环，直到curr指向尾节点

curr=curr.getNext();

}

curr.setNext(node);//设定尾节点后继为node

node.setPrior(curr);//设定node前序为尾节点

}

/\*\*

\* 在curr后插入结点node

\* @param curr 现在指向的节点

\* @param node 要插入的节点

\*/

public static void insert(Node curr,Node node){

if (curr == null) {

System.out.println("给定节点为空");

return;

}

if (node == null) {

System.out.println("给定节点为空");

return;

}

Node next = curr.getNext();

curr.setNext(node);

node.setPrior(curr);

node.setNext(next);

next.setPrior(node);

}

/\*\*

\* 删除节点 若删除头节点，返回新的头节点

\* @param node 要删除的节点

\*/

public static Node deleteNode(Node node){

if (node == null) {

System.out.println("给定节点为空");

return null;

}

if(node.getPrior() == null && node.getNext() == null){

System.out.println("编号为"+node.getNum()+"的节点已删除");

return node;

}

if (node.getPrior() == null) {//若node为头节点

Node next = node.getNext();

next.setPrior(null);

System.out.println("编号为"+node.getNum()+"的节点已删除");

return next;

}

if (node.getNext() == null) {//若node为尾节点

Node prior = node.getPrior();

prior.setNext(null);

System.out.println("编号为"+node.getNum()+"的节点已删除");

return null;

}

Node prior = node.getPrior();

Node next =node.getNext();

prior.setNext(next);

next.setPrior(prior);

System.out.println("编号为"+node.getNum()+"的节点已删除");

return null;

}

/\*\*

\* 更改node中数据

\* @param node

\* @param num

\*/

public static void update(Node node ,int num){

node.setNum(num);

}

/\*\*

\* 通过数字查num找节点

\*

\* @param head

\* @param num

\* @return 若找到返回节点，若未找到返回null

\*/

public static Node getNodeByNum(Node head,int num){

if (head == null) {

System.out.println("给定链表为空");

return null;

}

Node curr = head;

while (curr.getNum()!=num){

curr=curr.getNext();

}

if(curr.getNum()==num)return curr;

//if(curr.getNext()==null)return null;

return null;

}

}

Test.java中解决约瑟夫问题

public class Test {

public static void main(String[] args) {

Node head = new Node(1);

Node curr = head;

int n = 30;//此为人数，暂定为30

int m = 3;

for(int i = 2;i<=n;i++){

Node.increase(curr,new Node(i));

curr = curr.getNext();

}

int k=0;

curr = head;

while (n>0){

k++;

if(k>m)k=1;

if(k==m){

Node temp;

if ((temp=Node.deleteNode(curr)) != null) {

if(temp==curr){head=null;}

head = temp;

}

n--;

}

if(curr.getNext()==null){

curr=head;

}else {

curr=curr.getNext();

}

}

}

}