КАЗАНСКИЙ(ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

Семестровая работа №1
по алгоритмам и структурам данных
“Circle List”
Вариант 10

Выполнил студент группы 11-906:
Гарайшин Руслан
Проверил:
старший преподаватель кафедры
программной инженерии,к.ф.-м.н
Хадиев К.Р.

Казань 2020

Задача

Класс CircleList - класс, хранящий кольцевой список из N участников какой-нибудь игры некоторой игры. Каждый участник (класс Participant) хранит в себе два поля - имя и пол. Кольцевой список – next у последнего элемента является head.
Реализовать методы:

- Конструктор CircleList(String filename): создание кольцевого списка с помощью считывания информации из текстового файла.
- void show(): вывод содержимого спискового файла на экран;
- void insert(Participant p): вставка нового участника в список;
- void delete(String name): удаление участника из списка по имени;
- void sort(String name): выбрав некоторый элемент с именем name в качестве начального, провести сортировку исходного списка
- Participant last(int k): начав отчет с первого, удаляя каждого k-го. Установить последнего оставшегося участника
- CircleList [] gender(): из списка участников построить два списка, состоящих соответственно из мужчин и женщин. Метод возвращает ссылки на соответствующие подсписки.

Алгоритм

Для начала создаем класс Participant-класс участника игры.У каждого участника есть имя и пол.В конструктор участника подается отдельная строчка из файла с именем и полом,которая разбивается на 2 части с помощью StringTokenizer. У участника есть поле next-ссылка на след элемент в списке.Каждое поле приватно и имеет get/set-методы для доступа из других классов,также существуют 2 переопределенных метода toString() и equals(String name) для вывода содержимого участника и сравнивания 2 объектов.

```
public class Participant {
   private Participant next;
   private String name;
   private String gender;
   public Participant(String participantLine){
       StringTokenizer st=new StringTokenizer(participantLine);
       this.name=st.nextToken();
       this.gender=st.nextToken();
   }
public String getName(){
   return name;
}
public String getGender(){
   return gender;
}
public void setNext(Participant next){
   this.next=next;
}
public Participant getNext(){
   return next;
}
@Override
public boolean equals(Object obj){
   if(!(obj instanceof Participant)) {
       return false;
   }
   Participant Participant1=(Participant) obj;
   return(gender==Participant1.getGender() && name==Participant1.getName());
}
@Override
public String toString(){
   return name + " " + gender;
}
```

Класс CircleList

У кругового списка CircleList существует 2 поля head и tail,для них созданы get/set-методы.Конструктор кругового списка принимает в себя String

переменную-название файла(который находится в в папке src),из которой идет считывание участников в сам кольцевой список с помощью BufferedReader и самописного метода класса CircleList insert();.

```
public class CircleList {
  private  Participant head;
  private  Participant tail;
  private int counter;
  public CircleList(String Text) throws IOException {
      BufferedReader in = Files.newBufferedReader(Paths.get("src\\"+Text));
      String participantLine=in.readLine();
      while(participantLine!=null){
          Participant New=new Participant(participantLine);
          insert(New);
          participantLine=in.readLine();
          counter++;
      }
      in.close();
  }
  public CircleList(){
  }

  public Participant getHead() {
      return head;
  }

  public Participant getTail() {
      return tail;
  }

  public  void insert(Participant p) {
      if (head == null) {
          head = p;
      } else {
          tail.setNext(p);
      }
      tail = p;
      tail.setNext(head);
  }
```

Метод insert() получает на вход класс Участника,которого он должен вставить в кольцевой список.Если в исходном кольцевом списке нет участников,то исходный элемент становится head и tail.В противном случае элемент вставляется в конец списка с изменением tail и ссылкой на след элемент. с помощью setNext();
Insert() сложность:по времени O(1) и памяти O(p+n)
Delete(String name) Удаляет из списка участника с искомым именем.

```
public void delete(String name) {
  Participant current = head;
```

```
  Participant temp=current;
      if(head!=null){
          if(current.getName().equals(name)){
              head=head.getNext();
              tail.setNext(head);
          }else{
              if(tail.getName().equals(name)){
                  tail=temp;
              }else{
                  while(temp!=tail){
                      if(current.getName().equals(name)){
                          temp.setNext(current.getNext());
                          break;
                      }
                      temp=current;
                      current=current.getNext();
                  }
              }
          }
      }
  }
```

Имеется 2 локальные переменные типа Participant current и temp.Сначала проверяется пустой ли список,если список пустой,далее идет проверка является ли искомый элемент head или tail,если да,то идет удаление обьекта и изменение ссылок в списке. Если элемент находится между head и tail ,то идет проверка по всем элементам в списке с помощью цикла while и переменной temp.Сложность по времени O(n+1+1)=O(n),по памяти O(name+n*participant).

```
public void show() {
  Participant current = head;
  Participant temp=current;
  if (head != null) {
      while (temp != tail) {
          System.out.print(current.toString()+"\n");
          temp=current;
          current=current.getNext();
      }
  }else{
      System.out.print("Список пустой"+"\n");
  }
  System.out.print("\n") ;
}
```

Функция show() выводит содержимое списка,полученного из файла
Сначала проверяется наличие в списке элементов с помощью Head!=null,если в списке есть жлементы,то с помощью временной переменной Temp алгоритм проходится по всем элементам и выводит их с помощью переопределенного метода toString()

Оценка по времени O(n+1)=O(n)(т.к while имеет сложность O(n)),оценка по памяти O(1)

```
public CircleList[] gender() throws IOException {
  CircleList women = new CircleList();
  int count=0;
  CircleList men =new CircleList();
  CircleList[] gender={men,women};
  Participant temp=head;
  Participant current=head;
  while (temp!=tail){
      count++;
      temp=current;
      current=current.getNext();
  }
  current=head;
  for(int i=0;i<count;i++){
      if(current.getGender().equals("Ж")){
          women.insert(new Participant(current.toString()));
      }else{
          men.insert(new Participant(current.toString()));
      }
      current=current.getNext();
  }
  men.show();
  women.show();
  return gender;
}
```

Метод gender() создает два подсписка men и women,в которые складываются элементы главного списка в зависимости от пола игрока.Сначала считывается кол-во человек в списке,далее идет сортировка и запись в подсписки с помощью getGender(),insert() и equals().Далее идет вывод 2 списков на экран и возвращение массива gender,в котором находтся 2 ссылки на списки women и men.
O(n+n+n+n+n)=o(4n)=o(n)-оценка по времени
o(n*participant+m*partcipant+w*participant)-оценка по памяти

```
public Participant last(int k) {
  int pos=0;
  for(int i=2;i<=counter;i++){
      pos=(pos+k)%i;
  }
  pos++;
  Participant current=head;
  for(int i=0;i<pos-2;i++){
      current=current.getNext();
  }
  return current;
```

}

Все участники пронумерованы от 0до n-1,началом является Pos=0,нужно искулючить pos+(k-1) элемент ,начальная позиция след итерации будет pos+k,но если номер исключенного элемента N-1,след начальная позиция итерации будет N,что выходит за пределы списка,поэтому мы берем остаток от деления на кол-во оставших элементов (pes+k)%count

В итоге круг уменьшается на 1 элемент,при этом номер уцелевшего элемента в круге размером N равняется номеру в получившимся круге N-1,предположим ,что это работает в обратную сторону,берем круг размером i =2 ,рассчитываем кто будет в начальной позиции в след раунде. i в так0м случае кол-во не удаленных,а оставшихся ,оно увеличивается с каждой итерацией до N.

Сложность по времени O(n+n)=O(n)

оценка по памяти O(n+k+pos+count);