**Bag:**

/\*\*

\*

\* @author Marie Larson

\* @version 2/3/2018

\*/

public interface Bag<E> {

//returns current count of items in bag

int CurrentCount();

//checks if bag is empty

boolean CheckEmpty();

//checks if bag is full

boolean CheckFull();

//adds an item to the bag and returns true when item is added and false if item is not added

boolean Adder(E num);

//removes a random item from and returns this item

E remove();

//removes the first occurence of a specific item from bag and returns true if successful and false if it fails.

boolean remove(E num);

//clears all items from bag

void clear();

//returns count of occurence of a specific item in bag

int CountOfOccurence(E num);

//checks if an item exists in a bag

boolean CheckExistence(E num);

//returns string representation of the contents of the bag

String toString();

//returns true if the contents of the two bags are equal.

boolean equals(Object o);

}

**ArrayBag:**

import java.util.Random;

/\*\*

\*

\* @author Marie Larson

\* @version 2/3/2018

\*/

public class ArrayBag<E> implements Bag<E>{

private E[] list;

private int count;

public ArrayBag() {

this.list = ( E[] ) new Object[50];

count = 0;

}

public ArrayBag(int size){

this.list = ( E[] ) new Object[size];

count = 0;

}

@Override

public int CurrentCount(){

return count;

}

@Override

public boolean CheckEmpty(){

if(count==0)

{return true;}

else

{return false;}

}

@Override

public boolean CheckFull(){

return list.length == count;

}

@Override

public boolean Adder(E num){

if(count==list.length){

E[] newList = (E[]) new Object[count\*2];

for(int i = 0;i<count;i++){

newList[i]=list[i];

}

list = newList;

}

list[count++] = num;

return true;

}

//Finish Later

@Override

public E remove(){

if(count!=0){

Random random = new Random();

int RandomInt = random.nextInt(count);

E num = list[RandomInt];

for(int i=RandomInt;i<count-1;i++){

list[i]=list[i+1];

}

count--;

return num;

}

else{

return null;

}

}

//

@Override

public boolean remove(E num){

E rmv = num;

for(int i=0;i<=count;i++){

if(list[i].equals(rmv)){

count--;

return true;

}

}

return false;

}

@Override

public void clear(){

this.list = null;

this.count=0;

}

@Override

public int CountOfOccurence(E num){

int freq = 0;

for(int i=0;i<count;i++){

if(num.equals(list[i])){

freq++;

}

}

return freq;

}

//finish later

@Override

public boolean CheckExistence(E num){

for(int i=0;i<count;i++){

if(num.equals(list[i])){

return true;

}

}

return false;

}

public E get(int i){

if(i <= count){

return list[i];

}

else{

throw new ArrayIndexOutOfBoundsException();

}

}

@Override

public String toString(){

String tmp=new String();

tmp=tmp+getClass().getName() + '@';

for(int i = 0;i<count;i++){

tmp+=list[i] + ":";

}

return tmp;

}

@Override

public boolean equals(Object o){

if(!(o instanceof ArrayBag)){

return false;

}

ArrayBag b = (ArrayBag) o;

if(this.count != b.count){

return false;

}

for(int i=0;i<count;i++){

if(this.list[i] != b.list[i]){

return false;

}

}

return true;

}

}

**LinkedBag:**

import java.util.Random;

/\*\*

\*

\* @author marie

\* @version 2/6/2018

\*/

public class LinkedBag<E> implements Bag{

SinglyLinkedList<E> list;

int count;

public LinkedBag() {

this.list = new SinglyLinkedList();

this.count=0;

}

@Override

public int CurrentCount(){

return count;

}

@Override

public boolean CheckEmpty(){

return (count == 0);

}

@Override

public boolean CheckFull(){

return false;

}

@Override

public boolean Adder(Object num){

list.addLast((E)num);

count++;

return true;

}

@Override

public E remove(){

Random random = new Random();

int num = random.nextInt(count);

int var=0;

for(int i=0;i<count; i++){

E first = list.removeFirst();

if(var==num){

count--;

return first;

}

list.addLast(first);

var++;

}

return null;

}

@Override

public boolean remove(Object num){

for(int i=0;i<count;i++){

E first = list.removeFirst();

if(num.equals(first)){

count--;

return true;

}

else{

list.addLast(first);

}

}

return false;

}

@Override

public void clear(){

for(int i = 0;i<count;i++){

list.removeFirst();

}

count=0;

}

@Override

public int CountOfOccurence(Object num){

int freq = 0;

E f;

for(int i = 0; i<count; i++){

f = list.removeFirst();

if(num.equals(f)){

freq++;

}

list.addLast(f);

}

return freq;

}

@Override

public boolean CheckExistence(Object num){

E e;

for(int i=0;i<count;i++){

e=list.removeFirst();

if(num.equals(e)){

return true;

}

list.addLast(e);

}

return false;

}

public E get(int i){

if(i < count){

E e;

int p=0;

for(int c =0; c<count;c++){

e=list.removeFirst();

if(p==i){

return e;

}

list.addLast(e);

}

}

return null;

}

@Override

public String toString(){

String value = getClass().getName()+"@";

E e;

for(int i =0; i<count;i++){

e=list.removeFirst();

value = value + e + ":";

list.addLast(e);

}

return value;

}

@Override

public boolean equals(Object o){

if(!(o instanceof LinkedBag)){

return false;

}

boolean comp = true;

LinkedBag b = (LinkedBag)o;

E eList;

E eBag;

for(int i = 0;i<count;i++){

eList = list.removeFirst();

eBag = (E)b.list.removeFirst();

if(!eBag.equals(eList)){

comp = false;

}

list.addLast(eList);

b.list.addLast(eBag);

}

return comp;

}

}

**Player:**

/\*\*

\*

\* @author Marie Larson

\* @version 2/6/2018

\*/

public class Player {

private String name;

private String position;

private int number;

public Player(){

}

public Player(String name, String position, int number){

this.name = name;

this.position = position;

this.number = number;

}

public String getName(){

return name;

}

public void setName(String name){

this.name = name;

}

public String getPosition(){

return position;

}

public void setPosition(String name){

this.position = position;

}

public int getNumber(){

return number;

}

public void setNumber(int number){

this.number = number;

}

public String toString(){

return getClass().getName() + "@" + name + ":" + position + ":" + number;

}

@Override

public boolean equals(Object o){

if(!(o instanceof Player)){

return false;

}

Player p = (Player) o;

return ((p.name.equals(this.name))

&& (p.position.equals(this.position))

&& (p.number == this.number));

}

}

**Client:**

/\*\*

\*

\* @author Marie Larson

\* @version 2/6/2018

\*/

public class SinglyLinkedList<E> {

//--------Nested Node Class-----------

private static class Node<E>{

private E num;

private Node<E> next;

public Node(E e, Node<E> n){

num = e;

next = n;

}

public E getNum(){

return num;

}

public Node<E> getNext(){

return next;

}

public void setNext(Node<E> n){

next = n;

}

}

//----------End nested Node class---------

private Node<E> heads = null;

private Node<E> tails = null;

private int size = 0;

public SinglyLinkedList(){}

//methods

public int siz(){

return size;

}

public boolean CheckEmpty(){

return size == 0;

}

public E first(){

if(CheckEmpty()) return null;

return heads.getNum();

}

public E last(){

if(CheckEmpty()) return null;

return tails.getNum();

}

public void addFirst(E e){

heads = new Node<>(e, heads);

if(size==0){

tails = heads;

size++;

}

}

public void addLast(E e){

Node<E> newest = new Node<>(e, null);

if(CheckEmpty()){

heads = newest;

}

else{

tails.setNext(newest);

tails = newest;

size++;

}

}

public E removeFirst(){

if(CheckEmpty()) return null;

E anw = heads.getNum();

heads = heads.getNext();

size--;

if(size == 0)

tails = null;

return anw;

}

}

**Output of Client:**

run:

Client Class:

Enter next player

player name

Tom

player position

quarterback

player number

12

team size: 1

Would you like to enter another player? enter Y/N.

Y

Enter next player

player name

Harold

player position

defense

player number

63

team size: 2

Would you like to enter another player? enter Y/N.

Y

Enter next player

player name

Bob

player position

Bench Warmer

player number

73

team size: 3

Would you like to enter another player? enter Y/N.

Y

Enter next player

player name

Nathan

player position

water boy

player number

80

team size: 4

Would you like to enter another player? enter Y/N.

Y

Enter next player

player name

John

player position

linebacker

player number

23

team size: 5

Would you like to enter another player? enter Y/N.

Y

Enter next player

player name

Nick

player position

kicker

player number

90

team size: 6

Would you like to enter another player? enter Y/N.

N

Removing Random football player from team

Add a new player format: Tj, quarterback, 56

size of team: 6

Remove Tj from team

Size of team after removing player: 5

Create a Course List

Dropping random class

Print Course List:

ECE 320

Engr 402

MATH 266

Engl 320

CSCI 161

Creating NDSU Women's Basketball Team

Enter Player Information:

Name

Jess

Position

point

Number

40

Size of team: 1

Would you like to enter another player? enter Y/N.

Y

Enter Player Information:

Name

Nancy

Position

bench warmer

Number

34

Size of team: 2

Would you like to enter another player? enter Y/N.

Y

Enter Player Information:

Name

Stacy

Position

guard

Number

22

Size of team: 3

Would you like to enter another player? enter Y/N.

Y

Enter Player Information:

Name

Bell

Position

Front

Number

7

Size of team: 4

Would you like to enter another player? enter Y/N.

Y

Enter Player Information:

Name

Carol

Position

back

Number

3

Size of team: 5

Would you like to enter another player? enter Y/N.

Y

Enter Player Information:

Name

Mindy

Position

bench warmer

Number

9

Size of team: 6

Would you like to enter another player? enter Y/N.

N

Remove a player from the team

Add a new player

Current size of team: 6

Removing Michelle from Team

Current size of team without Michelle: 6

END

BUILD SUCCESSFUL (total time: 8 minutes 16 seconds)

