/\*\*

\*

\* @author Steven Glasford

\*

\* The interface file for the program, methods can be found in the Scores class

\*/

public interface Bag<E> {

/\*\*

\*

\* @return return the current size of the bag

\*/

public int getCurrentSize();

/\*\*

\*

\* @return whether or not the bag is empty

\*/

public boolean isEmpty();

/\*\*

\*

\* @param thing a generic object that you want to add to the bag

\*/

public void add (Object thing);

/\*\*

\*

\* @param thing the object that you want to remove from the bag

\* @return whether or not the item was removed from the bag

\*/

public boolean remove (Object thing);

/\*\*

\*

\* @return the object that was removed

\*/

public E remove( );

/\*\*

\* Used to destroy a bag

\*/

public void clear( );

/\*\*

\*

\* @param thing the item that is being searched for in the bag

\* @return the number of times the object was found in the bag

\*/

public int getFrequencyOf(Object thing);

/\*\*

\*

\* @param thing a generic object that you are searching for in the bag

\* @return true if the item is in the bag, false if not

\*/

public boolean contains(Object thing);

/\*\*

\*

\* @return The contents of the bag as a string

\*/

public String toString( );

/\*\*

\*

\* @param o a generic object

\* @return whether the objects are equal

\*/

public boolean equals(Object o);

}

/\*\*

\*

\* @author Steven Glasford

\* @version January 31, 2019

\* @param <E> a generic placeholder

\*/

public class LinkedBag<E> implements Bag{

//initialize the empty list

SinglyLinkedList list;

int count;

//construct an empty list

/\*\*

\* A constructor for the class

\*/

public LinkedBag(){

list = new SinglyLinkedList();

count = list.size();

}

/\*\*

\*

\* @return true if the bag is empty

\*/

@Override

public boolean isEmpty() {return list.isEmpty();}

/\*\*

\*

\* @param thing something you want to add

\*

\* I want to add the thing at the end

\*/

@Override

public void add (Object thing){

list.addLast(thing);

}

//

/\*\*

\*

\* @return return the current size of the list

\*

\*/

@Override

public int getCurrentSize(){

return list.size();

}

//find the first instance of an object and destroy it

/\*\*

\*

\* @param thing an object that you want to remove

\* @return whether or not the object was removed from the list

\*/

@Override

public boolean remove (Object thing){

for (int i = 0; i < list.size(); i++){

//keep in memory the removed item, and remove it

Object e = list.removeFirst();

if (thing.equals(e))

thing = e;

else

list.addLast(e);

}

//if the list is fully traversed and not found return false

return false;

}

//randomly remove an entrant

/\*\*

\*

\* @return the object that was randomly removed

\*/

@Override

public Object remove() {

//get the random entrant that is going to be destroyed

int random = (int)(Math.random() \* list.size() + 1);

//this will be the object that is going to be removed, stored temporarily

Object thing = null;

//traverse the list, keeping order, and removing the item

for (int i = 0; i < list.size(); i++){

//keep in memory the removed item, and remove it

Object e = list.removeFirst();

if (i == (random))

thing = e;

else

list.addLast(e);

}

//return the object that was removed

return thing;

}

//destroy the list

/\*\*

\* destroy the bag

\*/

@Override

public void clear( ){

for (int i = 0; i < list.size(); i++)

list.removeFirst();

}

//go through the list and determine the frequency of an object

/\*\*

\*

\* @param thing an object that you want to search for

\* @return the number of times that the object was found in the bag

\*/

@Override

public int getFrequencyOf(Object thing){

int frequency = 0;

for (int i = 0; i < list.size(); i++){

//if found increase the frequency

if (thing.equals(list.first()))

frequency++;

//rotate the list to the next variable

list.addLast(list.removeFirst());

}

//return the frequency of the object

return frequency;

}

//determine if the list contains an object

/\*\*

\*

\* @param thing an object that you are looking for

\* @return returns whether or not the object is contained in the bag

\*/

@Override

public boolean contains(Object thing){

boolean truth = false;

for (int i = 0; i < list.size(); i++){

//keep in memory the removed item, and remove it

Object e = list.removeFirst();

if (thing.equals(e))

truth = true;

list.addLast(e);

}

//return false if never found

return truth;

}

/\*\*

\*

\* @return convert the bag into a string

\*/

@Override

public String toString( ){

String words = null;

words = "[ ";

//return an empty list if the list is empty

if (list.size() == 0){

return (words + "]");

}

//rotate the list to get the next item

for (int i = 0; i < list.size(); i++) {

//get the first variable

words = words + list.first() + ", ";

//send the first variable in the list to the back

list.addLast(list.first());

//remove the first variable after moving it to the back

list.removeFirst();

}

words += " ]";

return words;

}

//get the item at a particular location in the list

public Object get(int i){

Object thing = null;

//if the user tries to find a spot that doesn't exist return null

if (i > list.size())

return thing;

for (int j = 0; j < (list.size()); j++){

if (j == (i)){

thing = list.first();

}

//traverse the list to keep order

list.addLast(list.first());

list.removeFirst();

}

return thing;

}

}

/\*\*

\*

\* @author Steven Glasford

\* @version January 31, 2019

\*

\* A structure to contain data about the player

\*/

public class Player{

String name = null;

String position = null;

int jerseyNum = 0;

/\*\*

\*

\* @param name The name of the player

\* @param position The position of the player

\* @param jerseyNum ` The jersey number of the player

\*/

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

this.jerseyNum=jerseyNum;

this.position=position;

this.name=name;

}

/\*\*

\*

\* @return the name of the player

\*/

public String getName() {return name;}

/\*\*

\*

\* @return The position of the player

\*/

public String position() {return position;}

/\*\*

\*

\* @return Return the jersey number of the player

\*/

public int jersey() {return jerseyNum;}

/\*\*

\*

\* @return Return the contents of the player class into a string

\*/

public String toString() {

return name + ", " + position + ", " + jerseyNum + "\n";

}

}

/\*\*

\*

\* @author Steven Glasford

\* @version January 31, 2019

\*

\* A main client program used to control the program

\*/

import java.util.Scanner;

public class Client {

/\*\*

\*

\* @param args no arguments passed into the system

\*/

public static void main(String[] args) {

//create an object of type Arraybag

ArrayBag footballTeam;

footballTeam = new ArrayBag(2) {};

//initialize the keyboard

Scanner keyboard = new Scanner(System.in);

System.out.println("Please enter some information about 6 players from the NDSU football team");

//make a for loop asking for the player's information

for (int i = 0; i < 6; i++){

String name = null;

String position = null;

int jerseryNum = 0;

String jerseyTemp = null;

boolean stop = false;

Player teamMember;

//enter the player's name

System.out.println("Please enter the player name: ");

name = keyboard.nextLine();

//enter the player's position

System.out.println("Please enter the player's position: ");

position = keyboard.nextLine();

//enter the player's jersey number

System.out.println("Please enter the player's jersey number: ");

//make sure the entered value is an integer

while (!stop){

jerseyTemp = keyboard.nextLine();

try{

Integer.parseInt(jerseyTemp);

stop = true;

} catch(NumberFormatException e){

System.out.println("Sorry that is not acceptable, please enter an integer: ");

stop = false;

}

}

jerseryNum = (int)Integer.parseInt(jerseyTemp);

//save the variables into the player and into the bag

teamMember = new Player(name,position,jerseryNum);

footballTeam.add(teamMember);

}

//printout the whole team

System.out.println(footballTeam);

//randomly remove a player

footballTeam.remove();

//print out the list again

System.out.println(footballTeam);

//add another player

Player team9 = new Player("Dumb", "Dumber", 69);

footballTeam.add(team9);

//printout the list again

System.out.println(footballTeam);

//remove the player I just added

footballTeam.remove(team9);

//print out the list again

System.out.println(footballTeam);

//create an instance called course to store course ids of classes i am taking

ArrayBag courses = new ArrayBag(2) {};

//add my courses to the list

courses.add("CSC161");

courses.add("ENGL234");

//print out the courses I am currently taking this semester

System.out.println("Courses I am Taking this semester: ");

System.out.println(courses);

//randomly remove one of my classes

courses.remove();

//print out the array with having removed one of the courses

System.out.println("Courses I am taking with one randomly removed" );

System.out.println(courses);

//create an instance of LinkedBag that contains information about the

//NDSU women's basketball team

LinkedBag basketballTeam;

basketballTeam = new LinkedBag();

System.out.println("Please enter 6 players from the NDSU women's basketball team");

for (int i = 0; i < 6; i++){

String name = null;

String position = null;

int jerseryNum = 0;

String jerseyTemp = null;

boolean stop = false;

Player teamMember;

//enter the player's name

System.out.println("Please enter the player name: ");

name = keyboard.nextLine();

//enter the player's position

System.out.println("Please enter the player's position: ");

position = keyboard.nextLine();

//enter the player's jersey number

System.out.println("Please enter the player's jersey number: ");

//make sure the entered value is an integer

while (!stop){

jerseyTemp = keyboard.nextLine();

try{

Integer.parseInt(jerseyTemp);

stop = true;

} catch(NumberFormatException e){

System.out.println("Sorry that is not acceptable, please enter an integer: ");

stop = false;

}

}

jerseryNum = (int)Integer.parseInt(jerseyTemp);

//save the variables into the player and into the bag

teamMember = new Player(name,position,jerseryNum);

basketballTeam.add(teamMember);

}

//printout the whole team

System.out.println(basketballTeam);

//randomly remove a player

basketballTeam.remove();

//print out the list again

System.out.println(basketballTeam);

//add another player

basketballTeam.add(team9);

//printout the list again

System.out.println(basketballTeam);

//remove the player I just added

basketballTeam.remove(team9);

//print out the list again

System.out.println(basketballTeam);

}

}

run:

Please enter some information about 6 players from the NDSU football team

Please enter the player name:

Holden Hotchkiss

Please enter the player's position:

QB

Please enter the player's jersey number:

15

Please enter the player name:

Bruce Anderson

Please enter the player's position:

RB

Please enter the player's jersey number:

8

Please enter the player name:

Hunter Luepke

Please enter the player's position:

FB

Please enter the player's jersey number:

44

Please enter the player name:

Eric Bachmeier

Please enter the player's position:

WR

Please enter the player's jersey number:

29

Please enter the player name:

Matt Anderson

Please enter the player's position:

TE

Please enter the player's jersey number:

88

Please enter the player name:

Karson Schoening

Please enter the player's position:

C

Please enter the player's jersey number:

59

[ Holden Hotchkiss, QB, 15

, Bruce Anderson, RB, 8

, Hunter Luepke, FB, 44

, Eric Bachmeier, WR, 29

, Matt Anderson, TE, 88

, Karson Schoening, C, 59

, ]

[ Holden Hotchkiss, QB, 15

, Bruce Anderson, RB, 8

, Eric Bachmeier, WR, 29

, Matt Anderson, TE, 88

, Karson Schoening, C, 59

, ]

[ Holden Hotchkiss, QB, 15

, Bruce Anderson, RB, 8

, Eric Bachmeier, WR, 29

, Matt Anderson, TE, 88

, Karson Schoening, C, 59

, Dumb, Dumber, 69

, ]

[ Holden Hotchkiss, QB, 15

, Bruce Anderson, RB, 8

, Eric Bachmeier, WR, 29

, Matt Anderson, TE, 88

, Karson Schoening, C, 59

, ]

Courses I am Taking this semester:

[ CSC161, ENGL234, ]

Courses I am taking with one randomly removed

[ CSC161, ]

Please enter 6 players from the NDSU women's basketball team

Please enter the player name:

Anna Goodhope

Please enter the player's position:

F

Please enter the player's jersey number:

5

Please enter the player name:

Taylor Thunstedt

Please enter the player's position:

G

Please enter the player's jersey number:

11

Please enter the player name:

Sarah Jacobson

Please enter the player's position:

G

Please enter the player's jersey number:

12

Please enter the player name:

Macey Kvilvang

Please enter the player's position:

F

Please enter the player's jersey number:

14

Please enter the player name:

Autumn Ogden

Please enter the player's position:

F

Please enter the player's jersey number:

20

Please enter the player name:

Rylee Nudell

Please enter the player's position:

F

Please enter the player's jersey number:

21

[ Anna Goodhope, F, 5

, Taylor Thunstedt, G, 11

, Sarah Jacobson, G, 12

, Macey Kvilvang, F, 14

, Autumn Ogden, F, 20

, Rylee Nudell, F, 21

, ]

[ Anna Goodhope, F, 5

, Sarah Jacobson, G, 12

, Macey Kvilvang, F, 14

, Autumn Ogden, F, 20

, Rylee Nudell, F, 21

, ]

[ Anna Goodhope, F, 5

, Sarah Jacobson, G, 12

, Macey Kvilvang, F, 14

, Autumn Ogden, F, 20

, Rylee Nudell, F, 21

, Dumb, Dumber, 69

, ]

[ Anna Goodhope, F, 5

, Sarah Jacobson, G, 12

, Macey Kvilvang, F, 14

, Autumn Ogden, F, 20

, Rylee Nudell, F, 21

, ]

BUILD SUCCESSFUL (total time: 6 minutes 19 seconds)