// (3/5)Waiting Chairs Class

// Nick Pinero

// 4/23/15

// Creates waiting chairs for the

// customers who arrive at the

// barber shop. If all chairs are

// full then the customer just leaves

//imports packages

//class name

public class WaitingChairs{

//instance variables

//creates an int variable

//number of charis in the

//barber shop

private int numChairs;

//creates an int variable

//to keep track of how many

//customers are in the chairs

private int numCustomers;

//creates two int pointer

//variables to determine who

//the first customer

//customer to be served and

//who the last customer to be

//served is.

private int first;

private int last;

//creates a customer array

//which will allow the

//customers to fill the chairs

//at the time they arrive

//if there is a chair open.

private Customer[] wait;

//Constructors

//sets the initial values for

//the instance variables.

public WaitingChairs(){

numChairs = 3;

numCustomers = 0;

first = 0;

last = 0;

wait = new Customer [numChairs];

}

//determines whether or not the

//chairs are empty

public boolean isEmpty(){

return numCustomers == 0;

}

//determines whether or not the

//chairs are full and returns a

//boolean

public boolean chairsFull(){

return numCustomers == numChairs;

}

//adds a customer to the chairs

//if there is an empty one

public void fillChairs(Customer c){

//if the chairs aren't full

if(!chairsFull()){

//print out that a chair is available

System.out.println("A chair is available");

//print out the customer's name that sits

//in the waiting chairs

System.out.println("\t" + c.getCustomerName() + "

sits");

//addthe customer to the

//chairs that the last chair

//pointer is pointing to

wait[last] = c;

//increase last so it points to

//a different/the last chair

last = (last +1)%numChairs;

//increases the number of customers

numCustomers ++;

//otherwise

} else {

//print out that all chairs are full

System.out.println("All chairs are full");

//get the customer's name and print out

//that customer leaves.

System.out.println("\t" + c.getCustomerName() + "

leaves");

}

}

//removes a customer from the

//waiting chairs and returns that

//customer.

public Customer leaveChair(){

//creates a variable to temporarily

//the customer being removed/leaving

//the chairs

Customer item = wait[first];

//if the chairs aren't empty

if(!isEmpty()){

//moves the first chair pointer to

//the next chair/person to be serviced

first = (first+1)%numChairs;

//decreases the number of customers

numCustomers --;

//otherwise

} else{

//sets item to null because

//there is nothing to return

item = null;

//informs the user that there isn't anyone

//in the chairs

System.out.println("No one is in the chairs");

}

//returns the customer stored in the

//temporary variable

return item;

}

//returns the customer to

//be served next (at the top

//of the queue).

public Customer peek(){

return wait[first];

}

//displays all the customers

//in the array/waiting in the

//chairs in the order which they

//arrived

public void displayChairs(){

//prints out waiting chiars to the

//user so they know that they are seeing

//the customers in the waiting chair

System.out.println("\nWaiting Chairs");

//if the chairs aren't empty

if(isEmpty() == false){

//goes through and displays each customer

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

wait[(first+i)%numChairs].display();

}

//otherwise

} else {

//prints out that there is

//no one in the waiting chairs

System.out.println("\tThere is no one in the waiting

chairs");

}

}

}