REPORT OF PROJECT

(Berk Belcioğlu 140315027 – İsmail Baş 150315040)

Firstly, we have used linked list structure in this project because in accordance with our experience we have decided that this choice can be good for us so we can comprehend the linked list structure and we can learn using classes in java and we can use their concrete instances (objects) in other classes. This approach can help us make progress in Java.

Firstly, we should have written a Node class (we named it Item).In Item class we have defined fields and we set in constructor its object which is AnyType,we set its next which is Item<AnyType> it is also an Item object.We have used two constructor but important constructor is first constructor.We have not used the other constructor but nevertheless we also have written it.We have written its methods like setNext which sets its next and getObject which returns its instance.We also have defined its fields as public because we have used its fields in bag class.Now we can pass to TheBag class.

Firstly, in bag class we have defined size, distinctSize, tempItem and firstItem.TempItem is traveler pointer in class.FirstItem is first Item in class which we have added.The constructor and clear method calls doClear so fields are set.IsEmpty returns that the bag is empty or not.Now we will explain important two methods add and remove.

# Add

Firstly it takes AnyType i and its instance is set with name item.If size = 0 item was assigned to firstItem and tempItem then tempItem = firstItem because we could not change firstItem which is head of our list.distinctSize++ because firstItem have a different string or integer or object whatever it is.In else;

if(contains(i) == false)

distinctSize++;

Here, we have kept the distinctSize.It is easiest way to count distinctSize when adding and removing.

if(size == 1)

{

tempItem.setNext(item);

firstItem.setNext(item);

tempItem = item;

size++;

}

Here, we have set next of tempItem and firstItem because we have used firstItem in toString.If we have not set next of firstItem we got an error “Null Pointer Exception” in toString so we have used if statement for (size == 1).We have assigned item to tempItem because item is created again and again and tempItem must have travelled.This structure resembled to stack.

else

{

tempItem.setNext(item);

tempItem = item;

size++;

}

Here, the rule continues for tempItem for (size > 1).

We can pass to remove method.

Remove

In remove we have created item, temp, temp2.Because there are many control statements for control and we must have used temp, temp2.

if(size == 1)

{

temp2 = null;

temp = null;

firstItem = null;

size--;

distinctSize--;

}

else if(size >= 2)

{

temp = firstItem;

temp2 = temp.next;

}

else

{

temp = null;

temp2 = null;

}

Here we have assigned temp and temp2 in first part of remove method.Temp2 must be next of temp because when we remove any item, we must not corrupt the structure.Items must be connect properly each other.

So, temp follows temp2.

if(contains(i))

In this statement we have controlled that if this statement is true then we can pass to remove process but if it is not true, remove method must return false so remove method will not process and nothing will happen.

for(int j = 0; j < size - 1; j++)

{

if((temp.object == item.object) && (j == 0))

{

if(elementSize(i) == 1)

distinctSize--;

firstItem = firstItem.next;

j = size;

size--;

}

else if((j >= 0 && j < size - 2) && (temp2.object == item.object))

{

if(elementSize(i) == 1)

distinctSize--;

temp.setNext(temp2.next);

temp2 = null;

j = size;

size--;

}

else if(temp2.object == item.object)

{

if(elementSize(i) == 1)

distinctSize--;

temp2 = null;

tempItem = temp;

j = size;

size--;

}

if(size + 1 != j)

{

temp = temp.next;

temp2 = temp2.next;

}

Here, we have used for loop.We have made trick like if j = 0 because for first Item and last Item there must be special cases and this special cases also valid for middle items.We also have controlled that elementSize == 1 because if elementSize is equals to 1 then distinctSize must decreased.

We have used another trick here like j = size; because when j == size the loop stops.After j = size, size--, so we must have controlled that (size + 1) != j that program should not reach last assignments beacuse remove process ends in if-else statements.The connecting of items is easy anyway you can see that there are special cases for firstItem, middle items and tempItem(lastItem) when removing.

Size method returns size of bag.Now we can discuss elementSize method.

int iSize = 0;

Item<AnyType> item = new Item<AnyType>(i, null);

Item<AnyType> temp;

temp = firstItem;

for(int j = 0; j < size; j++)

{

if(item.object == temp.object)

{

iSize++;

}

temp = temp.next;

}

return iSize;

In this method new node is defined again which is named as temp.Then we have assigned firstItem to temp then we have controlled the object of item with all objects of items of list.Then temp = temp.next so firstItem was not corrupted.When the condition is true iSize is increased.Then we have returned iSize.

distinctSize method returns distinct size.We have already kept distinctSize when adding and removing.Also defining distinctSize as a field made our work easier otherwise we could not run the distinctSize method properly.

contains method returns (elementSize(i) >= 1) and this is very good way to run properly this method.Now we can explain last method (toString).

toString

Item<AnyType> x;

String rStr = "[ ";

x = firstItem;

for(int i = 0; i < size - 1; i++)

{

rStr += x.object + " ";

x = x.next;

}

if(size > 0)

rStr += x.object + " ";

rStr += "]";

return rStr;

In toString firstly we have created string “[ “ and we have kept firstItem temporarily on another item which is named as x.Then we have added and we have assigned x.object, rStr and “ “ to rStr till loop ends.When loop ends rStr += x.object + " "; again because the last object still remains.if(size > 0) because when size == 0 x.object is null and rStr += x.object + " "; statement is unnecessary.At the end of these we have assigned and added “]” and rStr to rStr which means that the set of objects was ended.Then we have returned rStr.

The project ends here and we have written two different test class for integer and string.You can test it yourself and you can see that project works properly.Also you can add different things in the test classes, it is up to you ☺.