Laborübung
Klasse: 1DHIF
Datum: 7.03.23

Lernstoff:

- **Instanziieren und initialisieren von unterschiedlichen Arrays**
- **Iterieren durch Arrays**
- **Manipulieren und tauschen von Arraydaten**

Erweitere die (in der letzten Übungen entwickelten) Klassen ***IntZahlenArray*** und ***BooleanArray*** um die ergänzten Methoden und teste sie in der main Methoden

IntZahlenArray
- zahlen[]:int - ran:Random
+ main(String[]:args) + <<constructor>> IntZahlenArray (anz:int) + <<constructor>> IntZahlenArray () + ausgeben() + summe():int + **tauscheKleinGross()** + **diff()** + **kleinerDurchschnitt():int[]**

Die Methode ***tauscheKleinGross()*** tauscht das kleinste Element mit dem größten Element.
Bsp.: folgender Array

19	20	70	53	42

... wird zu folgendem Array verändert:

70	20	19	53	42

Die Methode ***diff()*** zeigt auf dem Bildschirm die Differenz benachbarter Elemente des *zahlen* Arrays getrennt durch einen Beistrich an. Stell dabei sicher, dass die ermittelte Differenz der beiden Zahlen immer mit einer positiven Zahl dargestellt wird.
Bsp: bei folgendem Array

70	20	19	55	42

... wird auf dem Bildschirm folgendes ausgegeben: 50, 1, 36, 13

Die Methode ***kleinerDurchschnitt ()*** ermittelt jene Zahlen des *zahlen* Arrays, die kleiner als der Durchschnitt der Zahlen im *zahlen* Array sind (Verwende die Methode *summe()*) . Für diese Zahlen wird ein int-Array erstellt und dieser wird befüllt und zurückgegeben.
Bsp: bei folgendem Array

20	15	40	45	30

... ist der Durchschnitt (20+15+40+45+30)/5=30

➐aus diesem Grund wird folgender Array zurückgegeben:

20	15

Erweitere die Klasse ***BooleanArray*** …

BooleanArray
- werte[]:boolean - ran:Random
+ main(String[]:args) + <<constructor>> BooleanArray (anz:int) + <<constructor>> BooleanArray () + ausgeben() + summe():int **+ printTrueBloecke():int** **+ nachRechts()** **+ nachRechts(int:anz)** **+ nachLinks()** **+ nachLinks(int:anz)**

printTrueBloecke() gibt die boolean Elemente blockweise aus – pro Block wird die Anzahl gefolgt von dem Wert (*true* oder false) *ausgegeben*.
Beispiele: (t steht für *true* und f steht für *false*)

Inhalt des *werte* Arrays	Ausgabe der Blöcke
{f,f,f}	3 false
{f,f,t,f,t,t,t,f}	2 false – 1 true – 1 false – 3 true – 1 false
{f,f,t,f,t,t,t,f,t}	2 false – 1 true – 1 false – 3 true – 1 false – 1 true
{f,f,t,f,t,t,t,f,f,t,t,t,t,t,t,t,t,t,t}	2 false – 1 true – 1 false – 3 true – 2 false – 12 true

nachRechts() lässt alle Elemente des *werte* Arrays um eine Stelle nach rechts „wandern“. Dadurch würde das letzte Element auf Position *length* kommen und somit aus dem Array „hinauswandern“. Dieses Element muss am Anfang des Arrays wieder eingereiht werden.
Beispiel (t steht für true und f steht für false)

{f,f,t,f,t,t,t,f,t}➐ {t,f,f,t,f,t,t,t,f}
{f,f,t,f,t,t,t,f,f,t,t,t,t,t,t,t,t,t,t} ➐{t,f,f,t,f,t,t,t,f,f,t,t,t,t,t,t,t,t,t}

nachRechts(int anz) lässt mit Hilfe der Methode *nachRechts()* alle Elemente des werte Arrays um *anz* Stellen nach rechts „wandern“.

nachLinks() lässt alle Elemente des werte Arrays um eine Stelle nach links „wandern“. Dadurch würde das erste Element auf Position -1 kommen und somit aus dem Array „hinauswandern“. Dieses Element muss am Ende des Arrays wieder eingereiht werden.
Beispiel (t steht für *true* und f steht für *false*)

{f,f,t,f,t,t,t,f,t}➐ {t,f,f,t,f,t,t,t,f}
{f,f,t,f,t,t,t,f,f,t,t,t,t,t,t,t,t,t,t} ➐{t,f,f,t,f,t,t,t,f,f,t,t,t,t,t,t,t,t,t}

nachLinks (int anz) lässt mit Hilfe der Methode *nachLinks ()* alle Elemente des werte Arrays um *anz* Stellen nach links „wandern“.