**Algorithmique / Travaux pratiques**

S12 – Ensemble d’entiers et BitSet





07.05.2015

Adriano De Almeida Silva – T-1f

Alex Travasso – T-1f

1. **Voici ci-dessous le type abstrait « ensemble d’entiers »**

# *Classe BitSetOfShorts*

package s12;

import java.util.BitSet;

public class BitSetOfShorts {

BitSet bs;

static final short *LOW* = Short.*MIN\_VALUE*;

static final short *HIGH* = Short.*MAX\_VALUE*;

// ------------------------------------------------------------

static int indexFromElt(short e) {

if (e < 0) {

int val = Math.*abs*(e);

return Math.*abs*((val \* 2) - 1);

} else {

return e \* 2;

}

}

static short eltFromIndex(int i) {

if (i % 2 == 0) {

return (short) (i / 2);

} else {

return (short) (((i + 1) / 2) \* -1);

}

}

// ------------------------------------------------------------

public BitSetOfShorts() {

bs = new BitSet(); // or: new BitSet(1 + HIGH - LOW);

}

// ------------------------------------------------------------

public void add(short e) {

bs.set(*indexFromElt*(e), true);

}

public void remove(short e) {

bs.set(*indexFromElt*(e), false);

}

public boolean contains(short e) {

return bs.get(*indexFromElt*(e));

}

// ------------------------------------------------------------

public void union(BitSetOfShorts s) {

BitSetOfShortsItr itr = new BitSetOfShortsItr(s);

while (itr.hasMoreElements()) {

add(itr.nextElement());

}

}

public void intersection(BitSetOfShorts s) {

BitSetOfShortsItr itr = new BitSetOfShortsItr(this);

while (itr.hasMoreElements()) {

short e = itr.nextElement();

if (!s.contains(e)) {

remove(e);

}

}

}

// ------------------------------------------------------------

public int size() {

return bs.cardinality();

}

// ------------------------------------------------------------

public boolean isEmpty() {

return bs.length() == 0;

}

// ------------------------------------------------------------

public String toString() {

String r = "{";

BitSetOfShortsItr itr = new BitSetOfShortsItr(this);

if (isEmpty())

return "{}";

r += itr.nextElement();

while (itr.hasMoreElements()) {

r += ", " + itr.nextElement();

}

return r + "}";

}

public static void main(String[] args) {

BitSetOfShorts a = new BitSetOfShorts();

BitSetOfShorts b = new BitSetOfShorts();

short[] ta = { -3, 5, 6, -3, 9, 9 };

short[] tb = { 6, 7, -2, -3 };

int i;

for (i = 0; i < ta.length; i++) {

a.add(ta[i]);

System.*out*.println("" + a + a.size());

}

for (i = 0; i < tb.length; i++) {

b.add(tb[i]);

System.*out*.println("" + b + b.size());

}

a.union(b);

System.*out*.println("" + a + a.size());

}

}

# *Classe BitSetOfShortsItr*

package s12;

public class BitSetOfShortsItr {

BitSetOfShorts b;

int index = 0; int cpt = 0;

public BitSetOfShortsItr(BitSetOfShorts theSet) {

b = theSet;

cpt = theSet.size();

}

public boolean hasMoreElements() {

return cpt > 0;

}

public short nextElement() {

while (!b.contains((short) index)) {

index++;

}

short nextEl = (short) index;

cpt--;

index++;

return nextEl;

}

}

*sortie console du test*



# Voici ci-dessous la classe BitSet