* Java:
* Algorithm analysis:
  + Running time (note add coefficients of inequality to find C(n)
    - Show f(n) = 35n + 12 is O(n)
    - You need something in the form f(n)<=C(n)
    - Let n0 = 1
    - Let C=47

n> 0

n>1

12n>12

35n + 12n > 35n + 12

47n> f(n)

C=47

Hence, f(n) is O(n)

* + - Show f(n) =3n^2 + 5n + 8 is O(n^2)
    - Let n0=1
    - Let C=16 🡪 f(n)< 16n^2

Assume n>1

n^2>1

then:

5n^2>5n

8n^2>8

13n^2>5n+8

16n^2>3n^2 + 5n + 8

16n^2>f(n)

C=16

* Stacks and Queues:
  + Method to copy the contents of a Stack to a Queue. When you are finished, the stack sould have the same contents as it did originally.

Void copy(Stack from, Queue to){

Stack work = new Stack();

While(!from.isEmpty()){ //copy from stack to work +queue

Object X = from.pop();

to.enqueue(X);

work.push(X);

}

while(!work.isEmpty()){

from.push(work.pop());

}

}

* + Method to Copy the contents of an array list to an array

void copy(ArrayList<String> from, String[] to){

int i = 0;

for (String S: from){ //iterate through the ArrayList

to [i] = s;

i++;

}

}

OR

void copy(ArrayList<String> from, String[] to){

for (int i=0; i<from.size(); i++){ //iterate while i<length

to [i] = from.get(i);

}

}

* + Enqueue method for a Queue implemented (as a circular Queue) in an array

class CircularQueue<T>{

int front, rear; //rear points to the next available slot

T[] data;

* + - * void enqueue( T item){

data[rear]=item;

rear=rear+1;

if(rear>=data.length){

rear = 0;

}

}

}

* Arithmetic Expressions: