*Find the minimum in an array

1) Algorithm Min
Start
// is already filled with values
// we have an array with N elements
array[0...N-1]
min = +infinity
minIndex = -1
for i from 0 -> N-1:
  if array[i] < min:
    min = array[i]
    minIndex = i;
print<<"Minimum value is "<<min<<endl;
print<<"Index of minimum valua is "<<minIndex<<endl;
End

*Find deposits, withdraws and balance in an array of movements
*We have an array of int values, positive values are deposits
*negative values are withdraws

2)Account
Start
movements[0...N-1]
totalDeposit = 0;
totalWithdraw = 0;
maxDeposit = movements[0];
maxWithdraw = movements[0];
for i from 0 -> N-1:
  if (movements[i] > 0):
    totalDeposit += movements[i]
    if (movements[i] > maxDeposit):
      maxDeposit = movements[i]
  else:
    totalWithdraw += movements[i]
    if (movements[i] < maxWithdraw)
      maxWithdraw = movements[i]

balance = totalDeposit + totalWithdraw
print<<"Total deposits are "<<totalDeposit<<endl;
print<<"Total withdraws are "<<totalWithdraw<<endl;
print<<"Max deposit is "<<maxDeposit<<endl;
print<<"Max withdraw is "<<-maxWithdraw<<endl;
print<<"Your balance is "<<balance<<endl;
End

*Find the sum of even numbers in an array
*if you encounter 0 or negative number, break out of the loop and print sum

3) Algorithm SumEven
Start
// this is the array of numbers
// content inside the array is not visible, just take care of the Algorithm
numbers[0...N-1]
sumEven = 0
for i from 0 -> N-1:
  if numbers[i] <= 0:
    break out of the loop;
  else if numbers[i] % 2 == 1:
    continue to the next iteration
  // if we got here, that means numbers[i] is for sure even
  sum += numbers[i]
print<<"Sum of the even numbers in the array before 0 or any negative number is "
  <<sum<<endl;
End

Print a rectangle using ''
*Ask the user for the width of the rectangle
*Print the rectangle
****
****
****
****
4) Algorithm PrintRectangle
Start
Read width;
height = 4
for i from 1 -> height:
  for j from 1 -> width:
    print<<"*";
  print<<endl;
End

*Print the following pattern (looks like half of the Christman tree)
*
**
***
****
*****
******
****

5) Algorithm PrintHalfTree
```
Start
Read N
for i from 1 -> N:
   for j from 1 -> i:
      print<<"*"
   print<<endl;
```

* We have an array A of int values (size of array is large N)
We know for sure that each element in the array is between 1 and 9
We are going to find the frequencies for each element in the array A
then we are going to print frequencies of each element using '*'
For example
Element 3: *****
Element 5: **********
Element 6: ***
Element 8: **********

6) Algorithm FindFrequencies
```
// Declare new array called Frequencies[9] with 9 elements
// the index of each element will serve as the number
// value of the array at a specific index will serve as frequence
// for example if A = {1, 2, 1, 1, 2, 3, 8, 9, 2, 1}
// then Frequencies = {4, 3, 0, 0, 0, 0, 0, 1, 1}
// the first element of Frequencies is 4 which means
// Number 1 was found 4 times in the array A
Start
Frequencies[9]
for i from 1 -> N:
   Frequencies[A[i]]++
// print the histogram now
for i from 1 -> 9:
   if Frequencies[i] != 0:
      print << "Element "<<i<<" :"
      for j from 1 -> Frequencies[i]:
         print<<'*';
      print<<endl;
```

** STD from an array of int
the formula for STD is sqrt(Variance)
variance is average of squared differences from the mean
STD = sqrt(sum(pow(abs(x-mean), 2))/n)

7) Algorithm STD

```
A[1 ... N]
// suppose that we have a function that returns us the mean in an array
mean = getMean(A)
sum = 0
for i from 1 -> N:
   sum += (A[i] - mean) * (A[i] - mean)
variance = sum / N
// suppose we have a function that returns us the sqrt of a value
std = sqrt(variance)
print<<"Standard deviation is " << std <<endl
```