Sort Array by Parity

# Question

Given an array A of non-negative integers, return an array consisting of all the even elements of A, followed by all the odd elements of A.

You may return any answer array that satisfies this condition.

**Example 1:**

Input: [3,1,2,4]

Output: [2,4,3,1]

The outputs [4,2,3,1], [2,4,1,3], and [4,2,1,3] would also be accepted.

Note:

1 <= A.length <= 5000

0 <= A[i] <= 5000

# Pseudo Code

## v1.0

Create a Dynamic Array

Run the First For Loop from i = 0 to size

If it is not even

Run the Second For Loop from j = i to size

If Element encountered is even

Swap arr[i] and arr[j]

Transfer the Elements to Dynamic Array

Return the Dynamic Array

## 

## 

## 

## v2.0

Create a Dynamic Array

Declare and Initialize j and tempVariable to 0

Run the First For Loop from i = 0 to size

If even

Swap A[i] and A[j]

Increment j by one

Assign A[i] to ptr[i]

Assign A[j] to ptr[j]

Else

Assign A[i] to ptr[i]

Return ptr

# Source Code

## V 1.0 (Language C)

/\*\*

\* Note: The returned array must be malloced, assume caller calls free().

\*/

int\* sortArrayByParity(int\* A, int ASize, int\* returnSize){

\*returnSize = ASize;

int\* ptr = (int\*)malloc(ASize \* sizeof(int));

int tempVar = 0;

for(int i=0 ; i<ASize ; i++) {

if(A[i] % 2 != 0) {

for(int j=i ; j<ASize ; j++) {

if(A[j] % 2 == 0) {

tempVar = A[i];

A[i] = A[j];

A[j] = tempVar;

}

}

}

}

for(int i=0 ; i<ASize ; i++) {

ptr[i] = A[i];

}

return ptr;

}

## V 2.0 (Language C)

/\*\*

\* Note: The returned array must be malloced, assume caller calls free().

\*/

int\* sortArrayByParity(int\* A, int ASize, int\* returnSize){

\*returnSize = ASize;

int\* ptr = (int\*)malloc(ASize \* sizeof(int));

int j = 0, tempVariable = 0;

for(int i=0 ; i<ASize ; i++) {

if((A[i] % 2) == 0) {

tempVariable = A[i];

A[i] = A[j];

A[j] = tempVariable;

ptr[i] = A[i];

ptr[j] = A[j];

j++;

}

else

ptr[i] = A[i];

}

return ptr;

}