

1. A mileage counter is used to measure mileage in an automobile. A mileage counter looks something like this

0	5	9	9	8
---	---	---	---	---

The above mileage counter says that the car has travelled 5,998 miles. Each mile travelled by the automobile increments the mileage counter. Here is how the above mileage counter changes over a 3 mile drive.

After the first mile

0	5	9	9	9
---	---	---	---	---

After the second mile

0	6	0	0	0
---	---	---	---	---

After the third mile

0	6	0	0	1
---	---	---	---	---

A mileage counter can be represented as an array. The mileage counter

0	5	9	9	8
---	---	---	---	---

can be represented as the array

```
int a[ ] = new int[ ] {8, 9, 9, 5, 0}
```

Note that the mileage counter is "backwards" in the array, a[0] represents ones, a[1] represents tens, a[2] represents hundreds, etc.

Write a function named `updateMileage` that takes an array representation of a mileage counter (which can be arbitrarily long) and adds a given number of miles to the array. Since arrays are passed by reference you can update the array in the function, you do not have to return the updated array.

You do not have to do any error checking. You may assume that the array contains non-negative digits and that the mileage is non-negative

If you are programming in Java or C#, the function signature is
void updateMileage counter(int[] a, int miles)

If you are programming in C or C++, the function signature is
void updateMileage counter(int a[], int miles, int len) where len is the number of
elements in the array

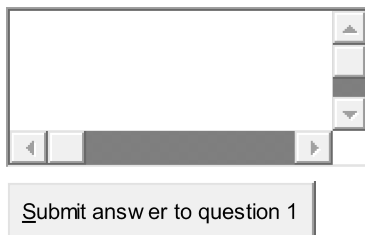
Examples:

if the input array is	and the mileage is	the array becomes
{8, 9, 9, 5, 0}	1	{9, 9, 9, 5, 0}
{8, 9, 9, 5, 0}	2	{0, 0, 0, 6, 0}
{9, 9, 9, 9, 9, 9, 9, 9, 9, 9}	1	{0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
{9, 9, 9, 9, 9, 9, 9, 9, 9, 9}	13	{2, 1, 0, 0, 0, 0, 0, 0, 0, 0}

Note that the mileage counter wraps around if it reaches all 9s and there is still some
mileage to add.

Hint: Write a helper function that adds 1 to the mileage counter and call the helper
function once for each mile

Copy and paste your answer here and click the "Submit answer" button



You should see a confirmation popup after hitting the submit button above. If you do
not see a confirmation popup, please [email](#) your answer.

2. An array is said to be *hollow* if it contains 3 or more zeros in the middle that are
preceded and followed by the same number of non-zero elements. Furthermore, all the
zeroes in the array must be in the middle of the array. Write a function
named *isHollow* that accepts an integer array and returns 1 if it is a hollow array,
otherwise it returns 0.

If you are programming in Java or C#, the function signature is
`int isHollow(int[] a)`

If you are programming in C or C++, the function signature is
`int isHollow(int a[], int len)` where `len` is the number of elements in the array

Examples:

if the input array is	is hollow?	reason
{1,2,0,0,0,3,4}	yes	2 non-zeros precede and follow 3 zeros in the middle
{1,1,1,1,0,0,0,0,0,2,1,2,18}	yes	4 non-zeros precede and follow the 5 zeros in the middle
{1, 2, 0, 0, 3, 4}	no	There are only 2 zeroes in the middle; at least 3 are required
{1,2,0,0,0,3,4,5}	no	The number of preceding non-zeros(2) is not equal to the number of following non-zeros(3)
{3,8,3,0,0,0,3,3}	no	The number of preceding non-zeros(3) is not equal to the number of following non-zeros(2)
{1,2,0,0,0,3,4,0}	no	Not all zeros are in the middle
{0,1,2,0,0,0,3,4}	no	Not all zeros are in the middle
{0,0,0}	yes	The number of preceding non-zeros is 0 which equals the number of following non-zeros. And there are three zeros "in the middle".

Hint: Write three loops. The first counts the number of preceding non-zeros. The second counts the number of zeros in the middle. The third counts the number of following non-zeros. Then analyze the results.

Copy and paste your answer here and click the "Submit answer" button



Submit answer to question 2

You should see a confirmation popup after hitting the submit button above. If you do not see a confirmation popup, please [email](#) your answer.

3. A positive number n is *consecutive-factored* if and only if it has factors, i and j where $i > 1$, $j > 1$ and $j = i + 1$. Write a function named **isConsecutiveFactored** that returns 1 if its argument is consecutive-factored, otherwise it returns 0.

the function signature is
`int isConsecutiveFactored(int n)`

Examples:

if n is	return	Because
24	1	$24 = 2 * 3 * 4$ and $3 = 2 + 1$
105	0	$105 = 3 * 5 * 7$ and $5 \neq 3 + 1$ and $7 \neq 5 + 1$
90	1	factors of 90 include 2 and 3 and $3 = 2 + 1$
23	0	has only 1 factor that is not equal to 1
15	0	$15 = 3 * 5$ and $5 \neq 3 + 1$
2	0	$2 = 1 * 2$, $2 = 1 + 1$ but factor 1 is not greater than 1
0	0	n has to be positive
-12	0	n has to be positive

Copy and paste your answer here and click the "Submit answer" button

Submit answer to question 3

You should see a confirmation popup after hitting the submit button above. If you do not see a confirmation popup, please [email](#) your answer.

Once you have completed all three answers and submitted them individually, you can safely close your browser.