

Problem 1

```
d={'data':1, 'structures':1000, 'and':15}

update(d, 'data', 1)

print(d) #expect: {'data':2, 'structures':1000, 'and':15}

update(d, 'and', 5)

print(d) #expect: {'data':2, 'structures':1000, 'and':20}

update(d, 'algorithms', 1)

print(d) #expect: {'data':2, 'structures':1000, 'and':20, 'algorithms':1}

update(d, 'algorithms', 4)

print(d) #expect: {'data':2, 'structures':1000, 'and':20, 'algorithms':5}
```

```
In [9]: def update(d, word, amount):
        if word in d:
            d[word] = d[word] + amount
        else:
            d.update({word: amount})
        return d

d = {'data': 1, 'structures': 1000, 'and': 15}
print(d)
print(update(d, 'data', 1))
print(update(d, 'and', 5))
print(update(d, 'algorithms', 1))
print(update(d, 'algorithms', 4))
```

```
{'data': 1, 'structures': 1000, 'and': 15}
{'data': 2, 'structures': 1000, 'and': 15}
{'data': 2, 'structures': 1000, 'and': 20}
{'data': 2, 'structures': 1000, 'and': 20, 'algorithms': 1}
{'data': 2, 'structures': 1000, 'and': 20, 'algorithms': 5}
```

Problem 2: A

<https://codingbat.com/prob/p170842>

Purpose: Returns a given string where for every char in the original, there are two chars.

Input: str, string input

Output: the original string with every char doubled

Assumptions: None

Given a string, return a string where for every char in the original, there are two chars.

```
double_char('The') → 'TThhee'  
double_char('AAbb') → 'AAAAbbbb'  
double_char('Hi-There') → 'HHii--TThheerree'
```

Go

...Save, Compile, Run (ctrl-enter)

Show Hint

```
def double_char(str):  
    new_str = ""  
    for char in str:  
        new_str = new_str + char + char  
    return new_str
```

Go

Expected	Run	
double_char('The') → 'TThhee'	'TThhee'	OK
double_char('AAbb') → 'AAAAbbbb'	'AAAAbbbb'	OK
double_char('Hi-There') → 'HHii--TThheerree'	'HHii--TThheerree'	OK
double_char('Word!') → 'WWoorrdd!!'	'WWoorrdd!!'	OK
double_char('!!!') → '!!!!'	'!!!!'	OK
double_char('') → ''	''	OK
double_char('a') → 'aa'	'aa'	OK
double_char('.') → '..'	'..'	OK
double_char('aa') → 'aaaa'	'aaaa'	OK
other tests		OK



All Correct

Good job -- problem solved. You can see our solution as an alternative.

See Our Solution

[next](#) | [chance](#)

Problem 2: B

<https://codingbat.com/prob/p119308>

Purpose: Determines if an array contains a 2 next to another 2

Input: nums, list of numbers

Output: returns true if the array contains a 2 next to another 2, otherwise false

Assumptions: None

Given an array of ints, return True if the array contains a 2 next to a 2 somewhere.

has22([1, 2, 2]) → True
has22([1, 2, 1, 2]) → False
has22([2, 1, 2]) → False

Go ...Save, Compile, Run (ctrl-enter)

```
def has22(nums):  
    for i in range(len(nums) - 1):  
        if nums[i] == 2 and nums[i + 1] == 2:  
            return True  
    return False
```

Go

Expected	Run	
has22([1, 2, 2]) → True	True	OK
has22([1, 2, 1, 2]) → False	False	OK
has22([2, 1, 2]) → False	False	OK
has22([2, 2, 1, 2]) → True	True	OK
has22([1, 3, 2]) → False	False	OK
has22([1, 3, 2, 2]) → True	True	OK
has22([2, 3, 2, 2]) → True	True	OK
has22([4, 2, 4, 2, 2, 5]) → True	True	OK
has22([1, 2]) → False	False	OK
has22([2, 2]) → True	True	OK
has22([2]) → False	False	OK
has22([]) → False	False	OK
has22([3, 3, 2, 2]) → True	True	OK
has22([5, 2, 5, 2]) → False	False	OK
other tests		OK

 All Correct

Problem 3

Choose one of the coding bat problems above and rewrite the solution using recursion. Submit a screen shot showing your recursive code and the results of running your code

In [8]: *#Purpose: Returns a given string where for every char in the original, there*
#Input: str, string input
#Output: the original string with every char doubled
#Assumptions: None

```
def double_char(str):  
    if len(str) == 1:  
        return str + str  
    else:  
        return str[0] + str[0] + double_char(str[1:])  
  
print(double_char('The'))  
print(double_char('AaBb'))  
print(double_char('Hi-There'))
```

TThhee
AAaaBBbb
HHii--TThheerree