Work with your neighbor. (This will be graded for participation only.)

1. Write a recursive function `join_all(alist)` that takes a list `alist` and returns a string consisting of every element of `alist` concatenated together.

   What is the base case?

   What is the recursive case?

2. Write a recursive function `join_all(alist,sep)` that takes a list `alist` and returns a string consisting of every element of `alist` concatenated together separated by the string `sep`.

   What is the base case?

   What is the recursive case?

3. Write a recursive function `sum_cols(grid, n)` that takes a list of lists of integers `grid` and integer `n` and returns the sum of column `n` in `grid`. For example, the call

```
sum_cols([[1,2,3,4], [10,20,30,40], [100,200,300,400]], 2)
```

should return `333`.

4. Write a recursive function `replace(s, a, b)` that takes as arguments the strings `s`, `a`, and `b` and returns a new string where all the occurrences of string `a` in string `s` are replaced with string `b`.

   a) For your solution, assume that string a is only one character long.

   b) For your solution, assume that string `a` can be of any size. Use `s.startswith(a)`, which returns `True` if string `s` starts with `a` and `False` otherwise.

5. Write a recursive function `get_even_positions(alist)` that returns a list consisting of the elements at the even-numbered positions of the Python list `alist` (the first element is at position 0). For example, the call

```
get_even_positions([2,6,5,33,8])
```

should return the list `[2,5,8]`.

Help: Sometimes it can be challenging to figure out what the base case(s) and recursive case(s) should be. Before you start writing the code, consider what should be returned for each of these calls:

```
get_even_positions([])              returns:
get_even_positions([2])             returns:
get_even_positions([2,6])           returns:
get_even_positions([2,6,5])         returns:
get_even_positions([2,6,5,33])      returns:
```

Now, before you write the base case(s) and the recursive case(s), answer these questions:

a) When the recursive call returns, what operation is performed and what types are the operands expected to be?

b) What is the type that is returned in the base case(s)?

Now write your solution.