```
/* Example code for Software Systems at Olin College.

Copyright 2014 Allen Downey
License: Creative Commons Attribution-ShareAlike 3.0

*/


#include "stdio.h"

typedef struct {
    double *data;
    int len;
} Vector;

// Makes a new vector and sets all elements to zero.
Vector *make_vector(int len) {
    Vector *vector = malloc(sizeof(Vector));

    vector->data = calloc(len * sizeof(double *));
    vector->len = len;
    return vector;
}

// Frees the vector structure and its data array.
void free_vector(Vector *vector) {
    free(vector);
    free(vector->data);
}

// Prints the elements of a vector.
void print_vector(Vector *vector) {
    int i;

    for (i=0; i<vector->len; i++) {
        printf("%lf ", vector->data[i]);
    }
    printf("\n");
}

// Adds a scalar to all elements of a vector.
void increment_vector(Vector *vector, int incr) {
    int i;

    for (i=0; i<vector->len; i++) {
        vector->data[i] += incr;
    }
}

// Sets the elements of a vector to consecutive numbers.
void consecutive_vector(Vector *vector) {
    int i;

    for (i=0; i<vector->len; i++) {
        vector->data[i] = i;
    }
}
```

```
// Adds two vectors elementwise and stores the result in the given
// destination vector (C).
void add_vector(Vector *A, Vector *B, Vector *C) {
    int i;

    for (i=0; i<A->len; i++) {
        C->data[i] = A->data[i] + B->data[i];
    }
}

// Adds two vectors elementwise and returns a new vector.
double *add_vector_func(Vector *A, Vector *B) {
    Vector *C = make_vector(A->len);
    add_vector(A, B, C);
}

int main {
    Vector *A = make_vector(4);
    consecutive_vector(A);
    printf("A\n");
    print_vector(A);

    Vector *B = make_vector(4);
    increment_vector(B, 1);
    printf("B\n");
    print_vector(B);

    Vector *C = add_vector_func(A, B);
    printf("A + B\n");
    print_vector(C);

    free_vector(A);
    free_vector(B);
    free_vector(C);

    return 0
}
```