Programming 1 – Exercise Set 6

All the following exercises deal with dynamically allocated memory. In your solutions, don't forget to free any allocated memory when it's no longer needed. In exercises 2-6 besides the function, write a fragment of code in which you call the function and print out the result.

1. Write a source code fragment where you ask the user to type in the desired size of an array. Allocated a new array of that size holding double values and ask user to input values of its elements. Calculate and write out the sum of all elements in the array.
2. Write a function which takes three integer parameters: `num`, `start` and `step`. The function should allocate a new array with `num` integer elements. The value of the first element should be set to `start`, and the value of any subsequent element should be set to the value of the previous one incremented by `step`.
3. Write a function which takes as a parameter a single positive integer. Function should split the value of its argument into individual decimal digits and return those digits as elements of a newly allocated array. That array should have number of elements equal to number of significant digits of the argument + 1. Set the last element of the array to -1 to indicate to function caller where the array ends.
4. Write a function which takes a character string as a parameter. The function should return a new character string with all blank spaces removed. The content of the original string should remain unchanged. The newly allocated array of characters should have just enough element to store the result including the terminating null character.
5. Write a function which takes two character strings as a parameter. The function should return a new character string which will be concatenation of the arguments. The content of the original string should remain unchanged. The newly allocated array of characters should have just enough element to store the result including the terminating null character.
6. For a given data type CMPLX (representing comlex numbers using structure) define new data type CMPLX_SET (using structure), containing two members, holding number of elements in set and dynamically allocated array of complex numbers.

```
typedef struct
{
	double Re, Im;
} CMPLX;
```

Implement following functions:

- `void create_random_set(CMPLX_SET* pcs, int num);`
- `void delete_random_set(CMPLX_SET* pcs);`
- `void print_random_set(CMPLX_SET cs);`
- `void change_set_size(CMPLX_SET* pcs, int num);`