Instructions:

1. This is a project that should be conducted by each student individually
2. Each student will answer orally to 5 questions related to his submission
3. Your submission consists of the following files: myfuncs.h , myfuncs.c , test.c , makefile
4. All type definitions, constant declarations and function prototypes should be in myfuncs.h
5. Implementations of all functions should be provided by myfuncs.c
6. test.c has a single main function that test all of your functions
7. You may define additional auxiliary functions to break up your implementation into smaller, easy-to-read blocks.

Part I:

Write 3 of the following functions in C (check assignment distribution):

```
1. char* trim(char* orig);
2. char* word_wrap(char* orig, int width);
3. char** explode(char* delimiter, char* orig, int* count);
4. char* implode(char* glue, char** pieces, int count);
5. char* str_shuffle (char*str);
6. char* make_pw(int min_size, int dig, int mix_cap);
```

Part II:

Define Node structure that contains an integer data and a next node pointer, and define Linked List structure that contains a head node pointer, then write 3 of the following functions (check assignment distribution):

```
1. void delete_from_i(linkedlist ll, int i);
2. int delete_duplicates(linkedlist ll);
3. linkedlist* rev_list(linkedlist ll);
4. void split_list(linkedlist ll, linkedlist* left, linkedlist* right);
5. void split_alternate(linkedlist ll, linkedlist* odd, linkedlist* even);
```

Part III:

Write a main function that reads input from the keyboard and tests all of the six function you wrote in the previous two parts. You can use the following function to print the elements of a linked list:

```
void print_list(linkedlist ll){
  if (ll) {
    node* p = ll.head;
    while(p){
      printf("%d-", p.data);
      P = p->next;
    }
    printf("|\n");
  }
}
```

Part IV:

Write a proper makefile that can be used to compile your project. Your makefile should contain that following targets:
All, Myfunc.o , test.o , CLEAN

Function Specifications

	Function	Description
I.1	`trim`	Returns a string that results from removing all white spaces from the orig string. Note: consult your notes for a list of white spaces
I.2	`word_wrap`	Returns a string that results from inserting newline characters in the orig string, to make the number of characters in each line less than or equal width, without breaking any word apart.
I.3	`explode`	Returns an array of strings, each of which is a substring of orig formed by splitting it on boundaries formed by the string delimiter.
I.3	`implode`	Returns a string that results from joining array peices elements of size count with a glue string.
I.4	`str_shuffle`	Returns a string that results from shuffling orig. One random permutation of all possible is created.
I.5	`make_pw`	Returns a string that results from concatenating count random alphanumeric characters. All passwords must contain at least one letter. If dig is 1 then you must include at least one digit in the result, if it is 0 then digits are optional. If mix_cap is 1 then you must mix letters of different cap cases. If it is 0 then mixing cases is optional.
II.1	`delete_from_i`	Deletes, properly, the node at position i from the linked list ll and returns nothing.
II.2	`delete_duplicates`	Removes all duplicates from the linked list ll and returns number of deleted nodes. ll is not necessarily ordered.
II.3	`rev_list`	Returns a linked list that has copies of all nodes of the linked list ll, but in the reversed order of how they appear in ll.
II.4	`split_list`	Copies all nodes of the 1st half of the linked list ll to the empty list left, and all nodes of the 2nd half to the empty list right and returns nothing. If count of nodes in ll is odd, make left the longer list
II.5	`split_alternate`	copies all nodes at odd positions of linked list ll to the empty list odd in same order, and the rest of the nodes to the empty list even in the reversed order, and returns nothing.

Student-Project Assignments

Student's ID	Part I	Part II
432107138	I.1 , I.3 , I.5	II.2 , II.3 , II.4
433101368	I.1 , I.4 , I.6	II.2 , II.4 , II.5
433101369	I.1 , I.3 , I.6	II.3 , II.4 , II.5
433103117	I.1 , I.4 , I.5	II.1 , II.2 , II.3
433104751	I.2 , I.3 , I.5	II.1 , II.2 , II.4
434101556	I.2 , I.4 , I.6	II.1 , II.2 , II.5
434102455	I.2 , I.3 , I.6	II.2 , II.3 , II.4
434104018	I.2 , I.4 , I.5	II.2 , II.4 , II.5
434104426	I.1 , I.3 , I.5	II.3 , II.4 , II.5
435101028	I.1 , I.4 , I.6	II.1 , II.2 , II.3
435101206	I.1 , I.3 , I.6	II.1 , II.2 , II.4
435102492	I.1 , I.4 , I.5	II.1 , II.2 , II.5
435102530	I.2 , I.3 , I.5	II.2 , II.3 , II.4
435103661	I.2 , I.4 , I.6	II.2 , II.4 , II.5
435104335	I.2 , I.3 , I.6	II.3 , II.4 , II.5
435105084	I.2 , I.4 , I.5	II.1 , II.2 , II.3
435105864	I.1 , I.3 , I.5	II.1 , II.2 , II.4
435106115	I.1 , I.4 , I.6	II.1 , II.2 , II.5
435106135	I.1 , I.3 , I.6	II.2 , II.3 , II.4
435106826	I.1 , I.4 , I.5	II.2 , II.4 , II.5
435107524	I.2 , I.3 , I.5	II.3 , II.4 , II.5
435107536	I.2 , I.4 , I.6	II.1 , II.2 , II.3
435108342	I.2 , I.3 , I.6	II.1 , II.2 , II.4
435108418	I.2 , I.4 , I.5	II.1 , II.2 , II.5
435108422	I.1 , I.3 , I.5	II.2 , II.3 , II.4
435108418	I.1 , I.3 , I.5	II.2 , II.3 , II.4
435108422	I.1 , I.4 , I.6	II.2 , II.4 , II.5

Student's ID	Part I	Part II
431102055	I.1 , I.4 , I.6	II.2 , II.4 , II.5
432104727	I.1 , I.3 , I.6	II.3 , II.4 , II.5
433105183	I.1 , I.4 , I.5	II.1 , II.2 , II.3
433107558	I.2 , I.3 , I.5	II.1 , II.2 , II.4
433108235	I.2 , I.4 , I.6	II.1 , II.2 , II.5
434100857	I.2 , I.3 , I.6	II.2 , II.3 , II.4
434100978	I.2 , I.4 , I.5	II.2 , II.4 , II.5
434101145	I.1 , I.3 , I.5	II.3 , II.4 , II.5
434101627	I.1 , I.4 , I.6	II.1 , II.2 , II.3
434102082	I.1 , I.3 , I.6	II.1 , II.2 , II.4
434102638	I.1 , I.4 , I.5	II.1 , II.2 , II.5
434103285	I.2 , I.3 , I.5	II.2 , II.3 , II.4
434106058	I.2 , I.4 , I.6	II.2 , II.4 , II.5
434107800	I.2 , I.3 , I.6	II.3 , II.4 , II.5
435100367	I.2 , I.4 , I.5	II.1 , II.2 , II.3
435103077	I.1 , I.3 , I.5	II.1 , II.2 , II.4
435103790	I.1 , I.4 , I.6	II.1 , II.2 , II.5
435103943	I.1 , I.3 , I.6	II.2 , II.3 , II.4
435104180	I.1 , I.4 , I.5	II.2 , II.4 , II.5
435104389	I.2 , I.3 , I.5	II.3 , II.4 , II.5
435105334	I.2 , I.4 , I.6	II.1 , II.2 , II.3
435106918	I.2 , I.3 , I.6	II.1 , II.2 , II.4
435108270	I.2 , I.4 , I.5	II.1 , II.2 , II.5
435108307	I.1 , I.4 , I.5	II.2 , II.3 , II.4