1. notable obstacles：

a) The algorithm of separator: how to rearrange the array without create a new array

b) what are the values to be returned in the cases where strings are empty

2. Test data:

a)int appendToAll(string a[], int n, string value);

string cast[6] = { "glenn", "carl", "carol", "rick", "maggie", "daryl" };

int j=appendToAll(cast, 6, “!!!”); check if it adds “!!!” to strings

int k=appendToAll(cast, -1, “!!!”); check if it returns -1

b)int lookup(const string a[], int n, string target);

string cast[6] = { "glenn", "carl", "carol", "rick", "maggie", "daryl" };

int j=lookup(cast, 6, “carol”); check if it can return the right the position

int k=lookup(cast, 6, “caRol”); check if it returns desired output if no match found

int i=lookup(cast,-1,”carol”); check if it returns -1 when it does not make sense

c)int positionOfMax(const string a[], int n);

string cast[6] = { "glenn", "carl", "carol", "rick", "maggie", "daryl" };

int k = positionOfMax(cast, 6); check if it can find the max

int k = positionOfMax(cast, 0); check if returns -1

d)int rotateLeft(string a[], int n, int pos);

string characters[5] = { "rosita", "bob", "sasha", "glenn", "michonne" };

int m = rotateLeft(characters, 5, 1); check if it rotates

int j = rotateLeft(characters, 5, 5); check if it returns -1 when pos is more than n-1

int k = rotateLeft(characters, 0, -1); check if it returns -1 when position is negative

e)int rotateRight(string a[], int n, int pos);

string characters[5] = { "rosita", "bob", "sasha", "glenn", "michonne" };

int p = rotateRight(characters, 5, 2); check if it rotates

int j = rotateRight(characters, 5, 5); check if it returns -1 when pos is more than n-1

int p = rotateRight(characters, 0,-1); check if it returns -1 when position is negative

f)int flip(string a[], int n);

string roles[6] = { "abraham", "tara", "", "daryl", "carol", "tyreese" };

int q = flip(roles, 4); check if it returns the right value

int p = flip(roles, -1); check if returns -1

g)int differ(const string a1[], int n1, const string a2[], int n2);

string roles[6] = { "abraham", "tara", "", "daryl", "carol", "tyreese" };

string group[5] = { "abraham", "tara", "tyreese", "", "maggie" };

int r = differ(roles, 6, group, 5);

int s = differ(roles, 2, group, 1); check if it works as desired

int g= differ(roles, 6, group, -2); check if it returns -1 if there is a negative n2

h)int subsequence(const string a1[], int n1, const string a2[], int n2);

string names[10] = { "sasha", "rick", "beth", "glenn", "bob", "michonne" };

string names1[10] = { "rick", "beth", "glenn" };

int t = subsequence(names, 6, names1, 3); check if it works as desired when there is a match

string names2[10] = { "sasha", "glenn" };

int u = subsequence(names, 5, names2, 2); check if it works as desired when there is no match

int v = subsequence(names, 5, names2, -1); check if it returns -1 if there is a negative n2

int w = subsequence(names, 5, names2, 6); check if it returns -1 when n2>n1 which does not make sense

i)int lookupAny(const string a1[], int n1, const string a2[], int n2);

string names[10] = { "sasha", "rick", "beth", "glenn", "bob", "michonne" };

string set1[10] = { "maggie", "bob", "glenn", "rick" };

int v = lookupAny(names, 6, set1, 4); check if it works as desired when there is a match

string set2[10] = { "daryl", "carol" };

int w = lookupAny(names, 6, set2, 2); check if it works as desired when there is no match

j)int separate(string a[], int n, string separator);

string cast[6] = { "maggie", "carl", "daryl", "rick", "michonne", "carol" };

int x = separate(cast, 6, "glenn"); check if it works as desired when separator is not one of the arrays

string cast2[4] = { "carol", "rick", "michonne", "daryl" };

int y = separate(cast2, 4, "daryl"); check if it works as desired when separator is one of the arrays