1. One of the largest obstacles I overcame was writing the function meld. I didn’t know how to once I had the two strings conjoined to put them in alphabetical order. What I ended up doing was doing something similar to the detect min and the place in front function. I ran some tests and they all passed.

b.

With the following strings I ran all these tests:

***string h[7] = { "moana", "tiana", "elsa", "ariel", "", "belle", "elsa" };***

***string g[5] = { "moana", "tiana", "ariel", "belle","ariel" };***

***string f[4] = { "belle", "ariel", "tiana", "elsa" };***

***string e[5] = { "belle", "belle", "belle", "tiana", "tiana" };***

***string t[7] = { "x","s","L","L","L","L","L" };***

***string w[5] = { "w", "w","e","e","e" };***

***string O[7] = { "moana", "tiana", "elsa", "ariel", "", "belle", "elsa" };***

***string x[4] = { "elsa", "elsa", "raya", "tiana" };***

***string y[4] = { "ariel", "belle", "elsa", "merida" };***

***string LL[3] = { "arnold","potato","snow" };***

***string lp[5] = { "valerie", "valeria","rae", "elizabeth", "annalise" };***

***string lp1[3] = { "anna","elizabeth","rae" };***

***string z[10];***

***string q[10];***

***string yy[10];***

***string yyy[10];***

if (countMatches(h, 7, "elsa") == 2)

cout << 1;

Checked if countMatches had 2 elsas

if (countMatches(h, 0, "elsa") == 0)

cout << 1.1;

Checked if countmatches is 0 in length that it will return 0

if (countMatches(h, 7, "") == 1)

cout << 2;

Checked if countMatches had 1 “”

if (countMatches(h, -1, "merida") == -1)

cout << 3;

Checked if countmatches will reutrn –1 since –1 is a bad argument

if (countMatches(h, 0, "elsa") == 0)

cout << 4 << endl;

Checked if countmatches will return 0 sine there is no elsa in an empty string

if (detectMatch(h, 7, "elsa") == 2)

cout << 5;

CChecked if detect match will find elsa in the third position

if (detectMatch(h, 0, "elsa") == -1)

cout << 5.1;

Checked if detect match will give out error since it cannot look for something in an empty string.

if (detectMatch(h, 2, "elsa") == -1)

cout << 6;

Checked if detect matches will return –1 since it couldn’t find elsa

if (detectMatch(h, 7, "coco") == -1)

cout << 6.1;

Checked if detect matches will return –1 since it couldn’t find coco

if (detectMatch(h, -10, "elsa") == -1)

cout << 6.2 << endl;

Checked if it will return –1 since –10 is a bad argument

if (detectSequence(h, 7, "koala", bg, en) == false)

cout << 7;

Checked if it will return false since koala is not in h

if (detectSequence(h, 0, "ariel", bg, en) == false)

cout << 7.1;

Checked if it will return false since ariel is not in the empty string h

if (detectSequence(h, 1, "elsa", bg, en) == false)

cout << 8;

Checked if it will return false since elsa is not in h[0]

if (detectSequence(h, 7, "ariel", bg, en) && bg == 3 && en == 3)

cout << 9;

Checked if it will return true since ariel is in h and it shows up in position 4

if (detectSequence(h, 7, "elsa", bg, en) && bg == 2 && en == 6)

cout << 10;

Checked if it will return true since elsa is in h and it shows up in position 3 and 7

if (detectSequence(h, -1, "elsa", bg, en) == false)

cout << 11 << endl;

Checked if it will return –1 since-1 is a bad argument.

if (detectMin(g, 4) == 2)

cout << 12;

Checked if the min is at place 3

if (detectMin(g, 2) == 0)

cout << 13;

Checked if it will return 0 since the min is at 0

if (detectMin(g, -1) == -1)

cout << 14;

Checked if it will return –1 since –1 is a bad argument.

if (detectMin(g, 5) == 2)

cout << 15;

Checked if it will return 2 since the min is at position 2

if (detectMin(g, 0) == -1)

cout << 16 << endl;

Checked if it will return –1 since there are no elements to examine.

if (moveToBack(g, 4, 1) == 1 && g[1] == "ariel" && g[3] == "tiana")

cout << 15;

Checked if it will return true since tiana was moved to the back and ariel was moved to pos 1

//if (moveToBack(g, 4, 2) == 2 && g[2] == "belle" && g[3] == "ariel")

cout << 16;

Checked if it will return true since belle is in pos 2 and ariel is at the back.

//if (moveToBack(g, -1, 2) == -1)

cout << 17;

Checked if it will return –1 since –1 is a bad argument.

if (moveToBack(g, 0, 2) == -1)

cout << 17.2;

Checked if it will return –1 since you cannot move an empty string.

//if (moveToBack(g, 4, -1) == -1)

cout << 18 << endl;

Checked if it will return –1 since –1 is a bad argument.

//if (moveToFront(f, 4, 2) == 2 && f[0] == "tiana" && f[2] == "ariel")

cout << 17;

Checked if it will return true since tiana was moved to the front and ariel was moved to pos 2

//if (moveToFront(f, 4, 3) == 3 && f[0] == "elsa" && f[3] == "tiana")

cout << 18;

Checked if it will return true since elsa was moved to the front and tiana is in pos 3

//if (moveToFront(f, 4, 1) == 1 && f[0] == "ariel" && f[1] == "belle")

cout << 19;

Checked if it will return true since ariel was moved to the front and belle is in pos 1.

//if (moveToFront(f, -1, 2) == -1)

cout << 20;

Checked if it will return –1 since –1 is a bad argument

if (moveToFront(f, 0, 2) == -1)

cout << 20.5;

Checked if it will return –1 since an empty string cannot be moved.

//if (moveToFront(f, 2, -10) == -1)

cout << 21 << endl;

Checked if it will return –1 since –10 is a bad argument.

if (detectDifference(h, 4, g, 4) == 2)

cout << 22;

Checked if it will return 2 since the first difference was at position 2.

if (detectDifference(h, 4, f, 4) == 0)

cout << 23;

Checked if it will return 0 since the first difference is at position 0

if (detectDifference(f, 4, g, 3) == 0)

cout << 24;

Checked if it will return 0 since the first difference Is at position 0

if (detectDifference(f, -4, g, 3) == -1)

cout << 24.5;

Checked if it will return –1 since –4 is a bad argument.

if (detectDifference(f, 0, g, 3) == -1)

cout << 24.6;

Checked if it will return –1 since one cannot compare to an empty string

if (detectDifference(f, 4, g, 0) == -1)

cout << 24.7;

Checked if it will return 01 since it cannot compare to empty string.

if (detectDifference(f, 4, g, -3) == -1)

cout << 24.75<<endl;

Checked if it will return –1 since –3 is a bas argument.

if (deleteDups(e, 5) == 2 && e[1] == "tiana")

cout << 25;

Checked if it will return true since duplicates were removed.

if (deleteDups(t, 7) == 3)

cout << 26;

Checked if it will return 3 since string is only 3 strings long.

if (t[2] == "L")

cout << 27;

Checked if it will return L since L should be at t[2]

if (deleteDups(w, -1) == -1)

cout << 27.5;

Checked if it will return –1 since it’s a bad argument

if (deleteDups(w, 5) == 2)

cout << 28;

Checked if it will return 2 since that’s the new size of the array

if (deleteDups(w, 0) == -1)

cout << 28.5;

Checked if it will return –1 since there cannot be duplicates in empty string

if (contains(h, 7, g, 2))

cout << 30;

Checked if it will return true since g is in h

if (contains(h, 7, f, 1))

cout << 31;

Checked if it will return true since f is in h

if (contains(h, 7, g, 3))

cout << 31.5;

Checked if it will return true since g is in h.

if (contains(h, 0, g, 0)==true)

cout << 31.52;

Checked if it will return since the empty is in the empty.

if (contains(h, 1, g, 0)==true)

cout << 31.53;

Checked if it will return true since the empty is in h

if (contains(h, 0, g, 1)==false)

cout << 31.54;

Checked if it will return false since there cant be anything in an empty string

if (contains(h, -1, g, 3)==false)

cout << 31.6;

Checked if it will return false since –1 is bad argument

if (contains(h, 1, g, -2)==false)

cout << 31.7;

Checked if it will return false since –2 is a bad argument.

if (!contains(h, 7, f, 2))

cout << 31.8 << endl;

Checked if it will return false since f is not in h

if (meld(x, 4, y, 4, z, 10) == 8 && z[5] == "merida")

cout << 32;

Checked if it will return true since the new array is 8 strings long and merida is pos 5

if (meld(x, 4, LL, 3, q, 10) == 7 && q[5] == "snow")

cout << 33;

Checked if it will return true since the new array is 7 long and snow is in pos 5.

if (meld(y, 4, LL, 3, yy, 10) == 7 && yy[5] == "potato")

cout << 34;

Checked if it will return true since the new array is 4 long and potato is in pos 5

if (meld(x, 0, y, 0, z, 10) == 0)

cout << 34.1;

Checked if it will return 0 since the new array is empty

if (meld(y, 4, lp, 5, yyy, 10) == -1)

cout << 34.2;

Checked if it will return –1 since lp is not in the right format

if (meld(y, 4, lp, 5, yyy, 3) == -1)

cout << 34.3;

Checked if it will return –1 since y plus lp are longer thaan yyy

if (meld(y, -1, lp, 5, yyy, 3) == -1)

cout << 34.4;

Checked if it will return –1 since –1 is a bad argument

if (meld(y, 3, lp, -3, yyy, 3) == -1)

cout << 34.5;

Checked if it will return –1 since –3 is a bad argument.

if (meld(y, 4, lp1, 3, yyy, 10) == 7 && yyy[2] == "belle")

cout << 35<<endl;

Checked if it will return true since new array is 7 long and belle is in pos 2.

if (split(O, 7, "elsa") == 3)

cout << 36;

Checked if it was split at pos 3

if (split(O, 7, "tiana") == 6)

cout << 37;

Checked if it was split at pos 6

if (split(O, -1, "tiana") == -1)

cout << 38;

Checked if it will return –1 since –1 is bad argument

if (split(O, 0, "tiana") == -1)

cout << 39;

Checked if it will return -1 since empty string cannot be split.