Here is my first batch of test cases from my prim program. If I get super bored, maybe I will write a RNG that will generate some harder test cases. Let me know if you guys get the same results.

**File Attachments**

1) testcases0.zip, downloaded 13 times

---

Welp... I did get bored. So here is some code for generating an edge description file for Project 3:

```c
int main() {

    srand((unsigned) time(NULL));
    FILE* fp = fopen("graph.txt", "w");
    int maxVert;
    int maxWeight;
    int lines;

    printf("Enter Max Vertex Value: ");
    scanf("%d", &maxVert);
    printf("Enter Max Weight Value: ");
    scanf("%d", &maxWeight);
    printf("Enter Number of Edge Descriptions: ");
    scanf("%d", &lines);

    for(int i = 0; i < maxVert; i++)
        fprintf(fp, "%d %d %d
", rand()%maxWeight, rand()%maxWeight, rand()%maxWeight);

    fclose(fp);

    return 0;
}
```

Modifying for your liking.
And I got really bored... Here are 10 test cases with output to compare.

Max Vertex Name = 100
Max Edge Weight = 100
# of Edge Descriptions = 100

Let me know if you get the same thing!

You're welcome.

File Attachments
1) testcases2.zip, downloaded 19 times

Subject: Re: TEST CASES
Posted by evtilley on Thu, 17 Nov 2016 16:52:06 GMT

I get very different results for graph1. I noticed that many of your nodes are not printing correctly--for example:

0 : 1;
1 : 51(0)48;

doesn't make much sense since 51's parent is 0. Here is my result for graph1:

0: 0;
1: 20(0)26, 30(0)14, 51(0)48;
2: 67(20)4, 14(30)50, 47(30)46, 48(30)7, 41(51)26;
3: 43(67)45, 99(47)19, 22(41)18;
4: 72(99)61, 75(99)29, 80(99)60, 83(99)22, 89(99)53, 16(22)52;
5: 96(75)98, 6(83)31, 29(83)22, 70(83)16, 11(89)32;
6: 7(6)10, 60(6)38, 61(29)81, 63(29)31, 17(11)28, 77(11)92;
7: 44(60)24, 21(63)54, 69(63)21, 10(77)59, 37(77)40;
8: 76(44)59, 94(69)60, 68(10)44;
9: 25(76)74, 40(76)40, 84(94)37, 4(68)47;
10: 27(25)67, 42(40)53, 15(84)81;
11: 18(27)3, 5(42)18, 23(42)55, 50(42)3, 73(15)16, 92(15)28;
12: 98(18)10, 64(5)77, 86(5)53, 52(23)84, 3(50)25, 28(73)8, 39(73)11, 55(92)98, 56(92)93;
13: 88(98)84, 36(86)47, 45(86)93, 26(52)65, 38(3)22, 71(3)86, 62(28)6;
14: 32(45)76, 58(38)85, 66(38)25, 33(62)62;
15: 1(32)74, 2(32)61, 90(66)60;
16: 13(1)69, 81(2)27, 82(90)3;
17: 65(13)3;
18: 12(65)29, 53(65)14, 59(65)24;
Did anybody else get what I got?

Subject: Re: TEST CASES
Posted by btlindow on Fri, 18 Nov 2016 02:29:20 GMT

Thanks for pointing that out! I found my error. My extreme value wasn't being set correctly and it was causing some lower key values to remain in the heap. My new result is now closer to what you got, minus the weight of my first spanning tree in the forest to be lower than yours. I also noticed that you are printing your levels wrong. Per the specs:

The vertex descriptions in a level are to be ordered by increasing vertex number.

Yours:

2: 67(20)4, 14(30)50, 47(30)46, 48(30)7, 41(51)26;

Also, in the example, there is a space between the level number and the colon that follows. This might be important if diff is used for grading.

0 : 0;
1 : 1(0)4, 3(0)2, 4(0)8, 5(0)3;
2 : 6(1)5, 8(4)7, 9(5)1;
3 : 7(6)11, 10(8)6;
weight: 47

But here is my new tree:

0 : 0;
1 : 20(0)26, 30(0)14, 51(0)48;
2 : 14(30)50, 41(51)26, 47(30)46, 48(30)7, 67(20)4;
3 : 22(41)8, 43(67)45, 99(47)19;
4 : 16(22)52, 72(99)48, 75(99)29, 80(99)60, 83(99)22, 89(99)53;
5 : 6(83)31, 11(89)32, 29(83)22, 70(83)16, 96(75)98;
6 : 7(6)10, 17(11)28, 60(6)38, 61(29)81, 63(29)31, 77(11)92;
I will try to find the discrepancy for the weights being different.

Thanks for the feedback! I wouldn't have found that if you didn't point it out.

---

Subject: Re: TEST CASES
Posted by btlindow on Fri, 18 Nov 2016 02:42:10 GMT

I found it. In line 4 of the first spanning tree, you included the vertex/edge description:

72(99)61

And mine is:

72(99)48

If you look in graph1.txt you can see that on lines 14 and 30, we have a repeating edge description:

14: 99 72 48 ;
30: 99 72 61 ;

I am not sure if your table or matrix or adjacency list is being updated correctly. This caused our weights to be off by this exact amount. I didn't look any further than line 4 of our trees after I found
it. Hope this helps.

Ben

Subject: Re: TEST CASES
Posted by btlindow on Fri, 18 Nov 2016 03:32:30 GMT
View Forum Message <> Reply to Message

The previously posted test cases proved to have a fault. Here is third batch of hopefully accurate test cases. They are the same files from testcases2.zip, but with different output files.

Sorry for spamming the thread.

File Attachments
1) testcases3.zip, downloaded 12 times

Subject: Re: TEST CASES
Posted by evtilley on Fri, 18 Nov 2016 03:58:04 GMT
View Forum Message <> Reply to Message

Thanks for the observations-- I made a few quick changes and we now get the exact same result on graph1. I will try the other test cases soon.

Subject: Re: TEST CASES
Posted by evtilley on Fri, 18 Nov 2016 06:17:38 GMT
View Forum Message <> Reply to Message

I got slightly different results on graph2. Here’s the diff: https://www.diffchecker.com/MheGYmya

Subject: Re: TEST CASES
Posted by btlindow on Fri, 18 Nov 2016 06:50:49 GMT
View Forum Message <> Reply to Message

Your weight is higher than mine by 83. I see you chose some higher edges than I did:

Me:
10(35)75
35(13)0

You:
10(33)83 (+8)
35(10)75 (+75)
Are you updating the key value when you come across a visited (not MST added) vertex?

Subject: Re: TEST CASES  
Posted by evtilley on Fri, 18 Nov 2016 07:49:30 GMT  
View Forum Message <> Reply to Message

I found the issue: your random edge generator is creating edges with a weight of zero, which is not allowed by the spec (it says that all edge weights are positive integers).

Subject: Re: TEST CASES  
Posted by btlindow on Fri, 18 Nov 2016 15:24:45 GMT  
View Forum Message <> Reply to Message

Ah! I see it now. I could have sworn that I read that an omitted weight would be counted as 0, giving me the idea that 0 is a valid edge weight. I see that the omitted values are now set to 1. I will go ahead and adjust the test cases and output... again....

Thanks for hashing this out with me!

Subject: Re: TEST CASES  
Posted by btlindow on Fri, 18 Nov 2016 15:42:29 GMT  
View Forum Message <> Reply to Message

I hate to keep doing this, but here is the fourth iteration of test cases. All of them are new. This time with non-zero edge weights.

File Attachments  
1) testcases4.zip, downloaded 11 times

Subject: Re: TEST CASES  
Posted by evtilley on Fri, 18 Nov 2016 19:33:25 GMT  
View Forum Message <> Reply to Message

Thanks for fixing that. On graph3, we get different results: https://www.diffchecker.com/CWTL7fg0

I think you may be choosing the repeated edge of lower weight, but we’re supposed to ignore subsequent repeated edges after the first.
Subject: Re: TEST CASES  
Posted by btlindow on Fri, 18 Nov 2016 20:26:15 GMT  
View Forum Message <> Reply to Message

I believe he told us in class that if there is a repeated edge description that we are to use the smallest of the repeats. I will ask for clarification in the specs thread.

Subject: Re: TEST CASES  
Posted by evtilley on Fri, 18 Nov 2016 21:02:53 GMT  
View Forum Message <> Reply to Message

I think he did say that in class, but the specs say

Quote: In order for your program to run on a randomly created graph, if an edge is given more than once, ignore subsequent occurrences. If a u, v edge is given, ignore subsequent u, v and v, u edges, since the presence of a u, v edge implies the presence of a v, u edge.

Subject: Re: TEST CASES  
Posted by lusth on Fri, 18 Nov 2016 21:43:44 GMT  
View Forum Message <> Reply to Message

I misspoke in class. The spec says ignore subsequent edges.

Subject: Re: TEST CASES  
Posted by lusth on Fri, 18 Nov 2016 21:49:00 GMT  
View Forum Message <> Reply to Message

I just eyeballed it, but it looks like I get the same output for graph1.txt. The weights are certainly the same.

Subject: Re: TEST CASES  
Posted by btlindow on Fri, 18 Nov 2016 22:22:53 GMT  
View Forum Message <> Reply to Message

Take 5.

File Attachments
1) testcases5.zip, downloaded 22 times
Subject: Re: TEST CASES  
Posted by evtilley on Fri, 18 Nov 2016 22:41:27 GMT  
View Forum Message <> Reply to Message

We now match on 8 of the 10 graphs. On graph 1, I ignore the loops in the input, which I think is acceptable. On graph 5, here is the comparison (yours in on the left): https://www.diffchecker.com/EP1ZAuBu

Subject: Re: TEST CASES  
Posted by lusth on Sat, 19 Nov 2016 01:39:55 GMT  
View Forum Message <> Reply to Message

Any intuition as to who is right on graph 5? My output agrees with one of yours. Since the total weight of the trees matches, could it have to do with breakking ties?

Subject: Re: TEST CASES  
Posted by evtilley on Sat, 19 Nov 2016 02:37:30 GMT  
View Forum Message <> Reply to Message

I tried commenting out my tie breaking logic, and I got the same output that btlindow got. So, either btlindow forgot to include that logic in his comparator, or mine is messed up.

Subject: Re: TEST CASES  
Posted by lusth on Sat, 19 Nov 2016 03:15:35 GMT  
View Forum Message <> Reply to Message

Iusth wrote on Fri, 18 November 2016 15:49 I just eyeballed it, but it looks like I get the same output for graph1.txt. The weights are certainly the same.

I take that back. I have differences. It's the tie-breaker for sure.

Subject: Re: TEST CASES  
Posted by btlindow on Sat, 19 Nov 2016 04:57:05 GMT  
View Forum Message <> Reply to Message

In my comparator, I am comparing the names of the vertices if there is a tie with they keys. I am going to try some testing to ensure it is working. Then comment it out to see how that output matches up.

Subject: Re: TEST CASES
Okay.... so here is what I observed in my tests. I made a really small test case that had the following edges (or very similar):

1 5 1 ;
1 2 1 ;
1 4 1 ;
1 9 1 ;
1 8 1 ;
1 28 1 ;

I was observing the extreme value of my binheap after each extractMin in Prim's. I noticed that my extreme values would store in ascending vertex number, that is:

b->extreme : vertex 1
extractMin
b->extreme : vertex 2
extractMin
b->extreme : vertex 4
extractMin
b->extreme : vertex 5
extractMin
b->extreme : vertex 8
extractMin
b->extreme : vertex 9
extractMin
b->extreme : vertex 28

This leads me to believe that my comparator is functioning correctly by choosing the vertex with the smaller node name if the key's are equivalent.

Next, I commented out my comparator's equivalency statement of:

/*if(x == y)
 {
  if(v->name < w->name) return -1;
  else return 1;
 }*/

Where x is vertex v's key value and y is vertex w's key value.

After doing this, running my Prim's, and comparing my output to yours... I too get exactly the same output as you (for graph 5).

Seeing that both of us have the equivalency statement in our comparator, I would assume it's safe to say that it's probably a glitch in the Matrix... (the Neo and Mr. Smith Matrix, not the adjacency Matrix.)
I am going to look closely at the test case and see what is happening.

---

**Subject: Re: TEST CASES**  
Posted by evtilley on Sat, 19 Nov 2016 06:43:01 GMT  
[View Forum Message] [Reply to Message]

I found a bug where I was caching vertex number incorrectly... oops. We now match on 10/10 of the inputs (except the loops in graph1.txt, which I just throw away).

---

**Subject: Re: TEST CASES**  
Posted by btlindow on Sat, 19 Nov 2016 06:50:42 GMT  
[View Forum Message] [Reply to Message]

Thats good to hear. I just spent the last half hour hand jamming this thing on paper and scratching my head. Almost threw my computer out the 7th floor window. Decided against it.

I am going to post the mother of all test cases sometime later with an absurd amount of vertices and edges. Probably within the next few minutes.

---

**Subject: Re: TEST CASES**  
Posted by evtilley on Sat, 19 Nov 2016 06:54:11 GMT  
[View Forum Message] [Reply to Message]

Sorry to put you through that. The zeroes in the input made me feel the same way yesterday-- it's so tough to debug when you don't know who's wrong!

Look forward to the next round of test inputs.

---

**Subject: Re: TEST CASES**  
Posted by btlindow on Sat, 19 Nov 2016 07:02:10 GMT  
[View Forum Message] [Reply to Message]

Don't even worry about it. Thats what this thread is for.

Here is 1000 descriptions, 1000 vertices, with up to a 1000 for weight.

We will start with this. If all goes well here, I will make a batch of them. It is pretty decently sized, so I am not sure if the diff website will work with it or not.

File Attachments  
1) graph.txt, downloaded 22 times
Subject: Re: TEST CASES  
Posted by evtilley on Sat, 19 Nov 2016 07:07:25 GMT

We should probably take this to PM to avoid spamming the forums.

Subject: Re: TEST CASES  
Posted by lusth on Sat, 19 Nov 2016 16:59:06 GMT

evtilley wrote on Sat, 19 November 2016 01:07 We should probably take this to PM to avoid spamming the forums.

No, this is exactly what the forum is for.

Subject: Re: TEST CASES  
Posted by evtilley on Sat, 19 Nov 2016 20:15:45 GMT

In that case, we matched on the 10K edges input. I had some performance issues, which I tracked down with the Microsoft Visual C++ profiler. It turned out that my adjacency matrix wrapper functions were responsible for 90% of the CPU time, so I inlined those and it now works in under 10 seconds.

Subject: Re: TEST CASES  
Posted by AdamJAlred on Mon, 21 Nov 2016 01:11:03 GMT

My output is matching both of yours, however, on the 10k input, output takes me roughly 30-50 seconds which isn't optimal. I imagine this will be a simple fix though.

Subject: Re: TEST CASES  
Posted by bpsokol on Tue, 22 Nov 2016 01:48:29 GMT

I disagree with graph1 from the 10 test cases above.
The initial difference is:
from the first page of the rubric the edge with the smallest initial vertex wins, and 23 is the smallest initial vertex of the two.

The only consideration would be if it is an eligible edge, however both of us both have 23 and 61 as eligible vertices in the prior line, so they are both eligible edges. I've attached my output as reference.

---

Are you saying it picks 95 over 23? I am confused by your wording.

This is what I observed.

We both agree that we have two equivalent keys in the top of our Priority Queue. We have Vertex 95 with a key value of 60 (which came from when 23 was added to the tree), and we have Vertex 64 with a key value of 60 (which came from when 61 was added to the tree).

Our comparator is supposed to pick the key value who belongs to the lower vertex number, in this case it would be Vertex 64 because: 64 < 95. So we add 64 to our tree and relax all of the edges that leave 64 and update those vertices that it touches. But after we pop 64 off the top, we know that 95 will be the next extreme on the top of the PQ.

Top of PQ: Vertex 95 : Key 60

Relax Edges from 64:
64 41 14 ;

(Potential***))Top of PQ: Vertex 41: Key 14

Relax Edge from 41:
95 41 58 ;
(Potential***Top of PQ: Vertex 95: Key 58

This is where it looks like we differ. When I added 64 to my tree, somewhere along the way I updated Vertex 95's key value to 58 - which was lower than the previous 60.

The reason I put Potential*** is because there could be other edges that are relaxed with lower key values that are added before the ones I listed. But we know that anything that is relaxed higher than 60 doesn't matter (per say) and anything lower than 60 will only be extracted before Vertex 95. I just pointed out the two vertices that would be updated before Vertex 95.

So, unless I am interpreting the rubric incorrectly... that is what I think is happening.

---

Subject: Re: TEST CASES
Posted by bpsokol on Tue, 22 Nov 2016 04:07:28 GMT

You are saying that 64 < 95, however those are the destination vertices. We are supposed to compare the source vertices first so 23 < 61, since 23 < 61 we pick 23 and never have to check if 64 < 95.

---

Subject: Re: TEST CASES
Posted by bpsokol on Tue, 22 Nov 2016 04:14:24 GMT

So I think the question is about how to interpret the rubric. It states:
Quote:If more than one edge is eligible to be added at any given point, chose the edge with the smallest vertex, breaking further ties with the smallest remaining vertex. For example, if 13 is the smallest edge weight among the remaining edges and there are these edges with weight 13 available:
3 5 13;
2 8 13;
2 9 13;
then the 2-8 edge would be added first, followed by the 2-9 edge, followed by the 3-5 edge.

If instead it was:
3 1 13;
2 8 13;
2 9 13;

What would the correct order be?
Subject: Re: TEST CASES
Posted by btlindow on Tue, 22 Nov 2016 04:44:13 GMT
View Forum Message <> Reply to Message

I can see what you are saying. And it makes sense. But the file contains edge descriptions for an undirected graph. So the first vertex in the line won't always be the "source" vertex and the second vertex won't always be the "destination" vertex.

For instance:

0 8 10 ;
1 8 13 ;
1 9 13 ;
2 9 13 ;

8 would be at the top of the PQ and then be the "source" for 1 when we relax it's edges. Then 1 would be at the top, update 9, and 9 would have a "source" of 1.

If there is a tie, are we to refer to the edge description file to see which vertex is the source? Or are we to consider the predecessor of a vertex to be the source?

I guess it comes down to the comparator, do we check the vertex's number or the number of the vertex's predecessor?

---

Subject: Re: TEST CASES
Posted by evtilley on Tue, 22 Nov 2016 04:57:01 GMT
View Forum Message <> Reply to Message

If Brian's interpretation is right, then we'd probably need to store (and update) the smallest incident vertex in our vertex structs so that the vertex comparator could access that value.

---

Subject: Re: TEST CASES
Posted by btlindow on Tue, 22 Nov 2016 05:00:21 GMT
View Forum Message <> Reply to Message

I was able to replicate bpsokol's output with a few added lines in the comparator. But I am still not 100% sure on the interpretation of the specifications. I guess I am going to page Dr. Lusth and see if he can't provide some input. Is bpsokol's output from graph1.txt "test1.actual.txt" correct or would my output1.txt be correct?

---

Subject: Re: TEST CASES
Posted by lusth on Tue, 22 Nov 2016 12:04:15 GMT
View Forum Message <> Reply to Message
To break ties:

- compare key values
  - if they are equal
    - compare predecessor numbers
      - if they are equal
        - compare vertex numbers

Subject: Re: TEST CASES
Posted by btlindow on Tue, 22 Nov 2016 16:40:18 GMT
View Forum Message <> Reply to Message

Test Cases 6.

Implemented the new found intuition for breaking ties.

How does this compare with what you guys get?

File Attachments
1) testcases6.zip, downloaded 16 times

Subject: Re: TEST CASES
Posted by lusth on Tue, 22 Nov 2016 17:00:00 GMT
View Forum Message <> Reply to Message

I get no differences for graph1.txt

Subject: Re: TEST CASES
Posted by maswain1 on Tue, 22 Nov 2016 23:06:23 GMT
View Forum Message <> Reply to Message

I am still confused by the order of the vertices. For graph1.txt it seems like the majority of the vertices are sorted by the vertex name. My comparator is breaking ties as posted above (and show in class today), but I am getting a different order for each level. It seems like my comparator is always choosing to decide the order based on key since that is the first thing to compare between the vertices.

This is the order my comparator is making comparisons currently:

result = x->key - y->key
if this these are equal
  result = x->pred->name - y->pred->name
if this these are equal too
  (then we use) result = x->name - y->name
return result

Am I missing something or misunderstanding what is meant by "vertex/pred numbers"?

---

Subject: Re: TEST CASES
Posted by nltollman on Wed, 23 Nov 2016 00:10:02 GMT
View Forum Message <> Reply to Message

Lusth provided the basic outline of the comparator function in the assign3 description. Here it is below.

```c
int vertexComparator(void *a, void *b)
{
    // cast from the generic type to the actual type
    vertex *v = (vertex *) a;
    vertex *w = (vertex *) b;

    // return a positive number if v's key > w's
    // return a zero if they are equal (break ties with vertex numbers)
    // return a negative number if v's key < w's
    ...
}
```

Basically, if v->key > w->key, a 1 is returned; if v->key == w->key, you break the tie, and Lusth mentioned earlier here how to do that (if all else fails you're gonna return a 0); finally, if v->key < w->key, return -1.

I hope this helps.

**Special note: this is the only time in the vertex class where we deal with void * parameters.

---

Subject: Re: TEST CASES
Posted by lusth on Wed, 23 Nov 2016 01:07:45 GMT
View Forum Message <> Reply to Message

maswain1 wrote on Tue, 22 November 2016 17:06 I am still confused by the order of the vertices. For graph1.txt it seems like the majority of the vertices are sorted by the vertex name. My comparator is breaking ties as posted above (and show in class today), but I am getting a different order for each level. It seems like my comparator is always choosing to decide the order based on key since that is the first thing to compare between the vertices.

This is the order my comparator is making comparisons currently:
result = x->key - y->key
if this these are equal
    result = x->pred->name - y->pred->name
if this these are equal too
    (then we use) result = x->name - y->name
return result

Am I missing something or misunderstanding what is meant by "vertex/pred numbers"?

The breaking ties bit is for finding the minimum values in the binomial heap. That's what the comparator does.

The ordering of the vertices in a level is done by vertex number (per the spec).

These are two different things.

---

Subject: Re: TEST CASES
Posted by maswain1 on Wed, 23 Nov 2016 16:50:47 GMT
View Forum Message <> Reply to Message

Ok, I see now. That cleared everything up, thank you!

---

Subject: Re: TEST CASES
Posted by jgmurphy1 on Sun, 27 Nov 2016 20:52:31 GMT
View Forum Message <> Reply to Message

SOLVED

This is about the test case given in the project 3 page
My output is for the first iteration of prim is:
0 : 0;
1 : 1(0)4, 3(0)2, 4(0)8, 5(0)3, 10(0)9;
2 : 6(1)5, 9(5)1, 8(10)6;
3 : 7(6)11;
when it should be
0 : 0;
1 : 1(0)4, 3(0)2, 4(0)8, 5(0)3;
2 : 6(1)5, 8(4)7, 9(5)1;
3 : 7(6)11, 10(8)6;
I think the mess up is happening from the check of vertices to relax is checking both directions so its changing 8(4)7 to 8(10)6 but it should be updating 10(0)9 to 10(8)6.
did anyone here have any hangups like this?
For the graphs in the 6th edition zip I get the same weights for all of them, but different graph outputs for graph1.txt and graph5.txt. I also get the same graph and weight for the 1k file

graph1 -
(mine)
15 : 12(61)1, 55(3)90, 64(61)60, 81(23)45, 83(92)59;
16 : 40(64)27, 41(64)14;
17 : 13(41)61, 22(41)73, 29(41)26, 95(41)58;
18 : 28(22)82, 69(29)55, 75(95)34, 97(95)26;
19 : 26(97)66, 76(97)30;
20 : 25(76)37;
21 : 60(25)21;

(yours)
15 : 12(61)1, 55(3)90, 81(23)45, 83(92)59, 95(23)60;
16 : 41(95)58, 75(95)34, 97(95)26;
17 : 13(41)61, 22(41)73, 26(97)66, 29(41)26, 64(41)14, 76(97)30;
18 : 25(76)37, 28(22)82, 40(64)27, 69(29)55;
19 : 60(25)21;

graph 5 -
(mine)
1 : 80(0)11, 95(0)42;
2 : 73(80)88, 78(95)7;
3 : 29(78)74, 51(78)11;
4 : 8(29)56, 18(51)50, 36(51)72, 99(51)18;
5 : 9(36)29, 15(8)24, 26(18)70, 33(36)50, 57(8)88, 82(18)26;
16 : 1(31)80, 38(31)76, 39(31)4, 85(4)59;
17 : 14(39)3, 19(38)27, 42(38)15, 48(38)73;
18 : 34(48)69, 61(19)26, 62(14)4, 64(14)63;
19 : 45(61)82, 63(34)13;

(yours)
1 : 48(0)76, 80(0)11, 95(0)42;
2 : 34(48)69, 38(48)73, 73(80)88, 78(95)7;
3 : 19(38)27, 29(78)74, 42(38)15, 51(78)11, 63(34)13;
4 : 8(29)56, 18(51)50, 36(51)72, 61(19)26, 99(51)18;
5 : 9(36)29, 15(8)24, 26(18)70, 33(36)50, 45(61)82, 57(8)88, 82(18)26;
16 : 1(31)80, 39(31)4, 85(4)59;
17 : 14(39)3;
18 : 62(14)4, 64(14)63;
I think that your output is the same as mine before I fixed my tiebreaker. Dr. Lusth confirmed that the current output1.txt is correct, so I would look there first.

Subject: Re: TEST CASES
Posted by jgmurphy1 on Tue, 29 Nov 2016 17:33:13 GMT

It was tie breaker. Matches exactly for 1 and 5 now. going to double check the rest

Subject: Re: TEST CASES
Posted by jgmurphy1 on Tue, 29 Nov 2016 17:43:07 GMT

I get the same for all tests in testcases6.zip and "the mother of all test cases"

Subject: Re: TEST CASES
Posted by sestephens on Tue, 29 Nov 2016 17:49:27 GMT

I match all of Ben's results for testcases6.zip