```cpp
/*---COP 3530 Project 1: Gator AVL---
 * Author: Benjamin Uppgard
 * Date: 09/29/2023
 *
 */

//#include "../src/main.cpp"
#define CATCH_CONFIG_MAIN
#include "catch.hpp"
#include "../src/GatorAVL.h"
#include <sstream>
#include <ostream>

/*
    To check output (At the Project1 directory):
        g++ -std=c++14 -Werror -Wuninitialized -o build/test test-unit/test.cpp && build/test
*/

TEST_CASE("BST Insert", "[flag]"){

        GatorAVL tree;   // Create a Tree object
        string name = "Bob";
        tree.InsertStudent(3, name);
        tree.InsertStudent(2, name);
        tree.InsertStudent(1, name);
        std::vector<unsigned int> actualOutput = tree.InOrder();
        std::vector<unsigned int> expectedOutput = {1, 2, 3};
        REQUIRE(expectedOutput.size() == actualOutput.size());
        REQUIRE(actualOutput == expectedOutput);

    REQUIRE(1 == 1);
}

TEST_CASE("BST Remove Case 1", "[flag]"){

    GatorAVL data;

    string name1 = "One";
    string name2 = "Two";
    string name3 = "Three";
    string name4 = "Four";
    string name5 = "Five";
    string name6 = "Six";
    string name7 = "Seven";
    string name8 = "Eight";
    string name9 = "Nine";
    string name10 = "Ten";
    string name11 = "Eleven";
    string name12 = "Twelve";
    string name13 = "Thirteen";
    string name14 = "Fourteen";
    string name15 = "Fifteen";
    string name16 = "Sixteen";

```

```
 55     data.InsertStudent(1, name1);
 56     data.InsertStudent(2, name2);
 57     data.InsertStudent(3, name3);
 58     data.InsertStudent(4, name4);
 59     data.InsertStudent(5, name5);
 60     data.InsertStudent(6, name6);
 61     data.InsertStudent(7, name7);
 62     data.InsertStudent(8, name8);
 63     data.InsertStudent(9, name9);
 64     data.InsertStudent(10, name10);
 65     data.InsertStudent(11, name11);
 66     data.InsertStudent(12, name12);
 67     data.InsertStudent(13, name13);
 68     data.InsertStudent(14, name14);
 69     data.InsertStudent(15, name15);
 70     data.InsertStudent(16, name16);
 71 
 72     data.Remove(1);
 73 
 74     std::vector<unsigned int> actualOutput = data.InOrder();
 75     std::vector<unsigned int> expectedOutput = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14
    , 15, 16};
 76     REQUIRE(expectedOutput.size() == actualOutput.size());
 77     REQUIRE(actualOutput == expectedOutput);
 78     REQUIRE(1 == 1);
 79 }
 80 
 81 TEST_CASE("BST Remove Case 2", "[flag]"){
 82 
 83     GatorAVL data;
 84 
 85     string name1 = "One";
 86     string name2 = "Two";
 87     string name3 = "Three";
 88     string name4 = "Four";
 89     string name5 = "Five";
 90     string name6 = "Six";
 91     string name7 = "Seven";
 92     string name8 = "Eight";
 93     string name9 = "Nine";
 94     string name10 = "Ten";
 95     string name11 = "Eleven";
 96     string name12 = "Twelve";
 97     string name13 = "Thirteen";
 98     string name14 = "Fourteen";
 99     string name15 = "Fifteen";
100     string name16 = "Sixteen";
101 
102     data.InsertStudent(1, name1);
103     data.InsertStudent(2, name2);
104     data.InsertStudent(3, name3);
105     data.InsertStudent(4, name4);
106     data.InsertStudent(5, name5);
107     data.InsertStudent(6, name6);
108     data.InsertStudent(7, name7);
```

```
109     data.InsertStudent(8, name8);
110     data.InsertStudent(9, name9);
111     data.InsertStudent(10, name10);
112     data.InsertStudent(11, name11);
113     data.InsertStudent(12, name12);
114     data.InsertStudent(13, name13);
115     data.InsertStudent(14, name14);
116     data.InsertStudent(15, name15);
117     data.InsertStudent(16, name16);
118 
119     data.Remove(3);
120 
121     std::vector<unsigned int> actualOutput = data.InOrder();
122     std::vector<unsigned int> expectedOutput = {1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14
    , 15, 16};
123     REQUIRE(expectedOutput.size() == actualOutput.size());
124     REQUIRE(actualOutput == expectedOutput);
125     REQUIRE(1 == 1);
126 }
127 
128 TEST_CASE("BST Remove Case 3", "[flag]"){
129 
130     GatorAVL data;
131 
132     string name1 = "One";
133     string name2 = "Two";
134     string name3 = "Three";
135     string name4 = "Four";
136     string name5 = "Five";
137     string name6 = "Six";
138     string name7 = "Seven";
139     string name8 = "Eight";
140     string name9 = "Nine";
141     string name10 = "Ten";
142     string name11 = "Eleven";
143     string name12 = "Twelve";
144     string name13 = "Thirteen";
145     string name14 = "Fourteen";
146     string name15 = "Fifteen";
147     string name16 = "Sixteen";
148 
149     data.InsertStudent(1, name1);
150     data.InsertStudent(2, name2);
151     data.InsertStudent(3, name3);
152     data.InsertStudent(4, name4);
153     data.InsertStudent(5, name5);
154     data.InsertStudent(6, name6);
155     data.InsertStudent(7, name7);
156     data.InsertStudent(8, name8);
157     data.InsertStudent(9, name9);
158     data.InsertStudent(10, name10);
159     data.InsertStudent(11, name11);
160     data.InsertStudent(12, name12);
161     data.InsertStudent(13, name13);
162     data.InsertStudent(14, name14);
```

```
163     data.InsertStudent(15, name15);
164     data.InsertStudent(16, name16);
165 
166     data.Remove(11);
167 
168     std::vector<unsigned int> actualOutput = data.InOrder();
169     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14
    , 15, 16};
170     REQUIRE(expectedOutput.size() == actualOutput.size());
171     REQUIRE(actualOutput == expectedOutput);
172     REQUIRE(1 == 1);
173 }
174 
175 TEST_CASE("BST Remove Case 4", "[flag]"){
176 
177     GatorAVL data;
178 
179     string name1 = "One";
180     string name2 = "Two";
181     string name3 = "Three";
182     string name4 = "Four";
183     string name5 = "Five";
184     string name6 = "Six";
185     string name7 = "Seven";
186     string name8 = "Eight";
187     string name9 = "Nine";
188     string name10 = "Ten";
189     string name11 = "Eleven";
190     string name12 = "Twelve";
191     string name13 = "Thirteen";
192     string name14 = "Fourteen";
193     string name15 = "Fifteen";
194     string name16 = "Sixteen";
195 
196     data.InsertStudent(1, name1);
197     data.InsertStudent(2, name2);
198     data.InsertStudent(3, name3);
199     data.InsertStudent(4, name4);
200     data.InsertStudent(5, name5);
201     data.InsertStudent(6, name6);
202     data.InsertStudent(7, name7);
203     data.InsertStudent(8, name8);
204     data.InsertStudent(9, name9);
205     data.InsertStudent(10, name10);
206     data.InsertStudent(11, name11);
207     data.InsertStudent(12, name12);
208     data.InsertStudent(13, name13);
209     data.InsertStudent(14, name14);
210     data.InsertStudent(15, name15);
211     data.InsertStudent(16, name16);
212 
213     data.Remove(13);
214 
215     std::vector<unsigned int> actualOutput = data.InOrder();
216     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14
```

```
216 , 15, 16};
217     REQUIRE(expectedOutput.size() == actualOutput.size());
218     REQUIRE(actualOutput == expectedOutput);
219     REQUIRE(1 == 1);
220 }
221
222 TEST_CASE("BST Remove Case 5", "[flag]"){
223
224     GatorAVL data;
225
226     string name1 = "One";
227     string name2 = "Two";
228     string name3 = "Three";
229     string name4 = "Four";
230     string name5 = "Five";
231     string name6 = "Six";
232     string name7 = "Seven";
233     string name8 = "Eight";
234     string name9 = "Nine";
235     string name10 = "Ten";
236     string name11 = "Eleven";
237     string name12 = "Twelve";
238     string name13 = "Thirteen";
239     string name14 = "Fourteen";
240     string name15 = "Fifteen";
241     string name16 = "Sixteen";
242
243     data.InsertStudent(1, name1);
244     data.InsertStudent(2, name2);
245     data.InsertStudent(3, name3);
246     data.InsertStudent(4, name4);
247     data.InsertStudent(5, name5);
248     data.InsertStudent(6, name6);
249     data.InsertStudent(7, name7);
250     data.InsertStudent(8, name8);
251     data.InsertStudent(9, name9);
252     data.InsertStudent(10, name10);
253     data.InsertStudent(11, name11);
254     data.InsertStudent(12, name12);
255     data.InsertStudent(13, name13);
256     data.InsertStudent(14, name14);
257     data.InsertStudent(15, name15);
258     data.InsertStudent(16, name16);
259
260     data.Remove(2);
261
262     std::vector<unsigned int> actualOutput = data.InOrder();
263     std::vector<unsigned int> expectedOutput = {1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14
    , 15, 16};
264     REQUIRE(expectedOutput.size() == actualOutput.size());
265     REQUIRE(actualOutput == expectedOutput);
266     REQUIRE(1 == 1);
267 }
268
269 TEST_CASE("BST Remove Case 6", "[flag]"){
```

```
270
271     GatorAVL data;
272
273     string name1 = "One";
274     string name2 = "Two";
275     string name3 = "Three";
276     string name4 = "Four";
277     string name5 = "Five";
278     string name6 = "Six";
279     string name7 = "Seven";
280     string name8 = "Eight";
281     string name9 = "Nine";
282     string name10 = "Ten";
283     string name11 = "Eleven";
284     string name12 = "Twelve";
285     string name13 = "Thirteen";
286     string name14 = "Fourteen";
287     string name15 = "Fifteen";
288     string name16 = "Sixteen";
289
290     data.InsertStudent(1, name1);
291     data.InsertStudent(2, name2);
292     data.InsertStudent(3, name3);
293     data.InsertStudent(4, name4);
294     data.InsertStudent(5, name5);
295     data.InsertStudent(6, name6);
296     data.InsertStudent(7, name7);
297     data.InsertStudent(8, name8);
298     data.InsertStudent(9, name9);
299     data.InsertStudent(10, name10);
300     data.InsertStudent(11, name11);
301     data.InsertStudent(12, name12);
302     data.InsertStudent(13, name13);
303     data.InsertStudent(14, name14);
304     data.InsertStudent(15, name15);
305     data.InsertStudent(16, name16);
306
307     data.Remove(6);
308
309     std::vector<unsigned int> actualOutput = data.InOrder();
310     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14
    , 15, 16};
311     REQUIRE(expectedOutput.size() == actualOutput.size());
312     REQUIRE(actualOutput == expectedOutput);
313     REQUIRE(1 == 1);
314 }
315
316 TEST_CASE("BST Remove Case 7", "[flag]"){
317
318     GatorAVL data;
319
320     string name1 = "One";
321     string name2 = "Two";
322     string name3 = "Three";
323     string name4 = "Four";
```

```
324     string name5 = "Five";
325     string name6 = "Six";
326     string name7 = "Seven";
327     string name8 = "Eight";
328     string name9 = "Nine";
329     string name10 = "Ten";
330     string name11 = "Eleven";
331     string name12 = "Twelve";
332     string name13 = "Thirteen";
333     string name14 = "Fourteen";
334     string name15 = "Fifteen";
335     string name16 = "Sixteen";
336 
337     data.InsertStudent(1, name1);
338     data.InsertStudent(2, name2);
339     data.InsertStudent(3, name3);
340     data.InsertStudent(4, name4);
341     data.InsertStudent(5, name5);
342     data.InsertStudent(6, name6);
343     data.InsertStudent(7, name7);
344     data.InsertStudent(8, name8);
345     data.InsertStudent(9, name9);
346     data.InsertStudent(10, name10);
347     data.InsertStudent(11, name11);
348     data.InsertStudent(12, name12);
349     data.InsertStudent(13, name13);
350     data.InsertStudent(14, name14);
351     data.InsertStudent(15, name15);
352     data.InsertStudent(16, name16);
353 
354     data.Remove(10);
355 
356     std::vector<unsigned int> actualOutput = data.InOrder();
357     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14
    , 15, 16};
358     REQUIRE(expectedOutput.size() == actualOutput.size());
359     REQUIRE(actualOutput == expectedOutput);
360     REQUIRE(1 == 1);
361 }
362 
363 TEST_CASE("BST Remove Case 8", "[flag]"){
364 
365     GatorAVL data;
366 
367     string name1 = "One";
368     string name2 = "Two";
369     string name3 = "Three";
370     string name4 = "Four";
371     string name5 = "Five";
372     string name6 = "Six";
373     string name7 = "Seven";
374     string name8 = "Eight";
375     string name9 = "Nine";
376     string name10 = "Ten";
377     string name11 = "Eleven";
```

```
378     string name12 = "Twelve";
379     string name13 = "Thirteen";
380     string name14 = "Fourteen";
381     string name15 = "Fifteen";
382     string name16 = "Sixteen";
383 
384     data.InsertStudent(1, name1);
385     data.InsertStudent(2, name2);
386     data.InsertStudent(3, name3);
387     data.InsertStudent(4, name4);
388     data.InsertStudent(5, name5);
389     data.InsertStudent(6, name6);
390     data.InsertStudent(7, name7);
391     data.InsertStudent(8, name8);
392     data.InsertStudent(9, name9);
393     data.InsertStudent(10, name10);
394     data.InsertStudent(11, name11);
395     data.InsertStudent(12, name12);
396     data.InsertStudent(13, name13);
397     data.InsertStudent(14, name14);
398     data.InsertStudent(15, name15);
399     data.InsertStudent(16, name16);
400 
401     data.Remove(4);
402 
403     std::vector<unsigned int> actualOutput = data.InOrder();
404     std::vector<unsigned int> expectedOutput = {1, 2, 3, 5, 6, 7, 8, 9,10, 11, 12, 13, 14
    , 15, 16};
405     REQUIRE(expectedOutput.size() == actualOutput.size());
406     REQUIRE(actualOutput == expectedOutput);
407     REQUIRE(1 == 1);
408 }
409 
410 TEST_CASE("BST Remove Case 9", "[flag]"){
411 
412     GatorAVL data;
413 
414     string name1 = "One";
415     string name2 = "Two";
416     string name3 = "Three";
417     string name4 = "Four";
418     string name5 = "Five";
419     string name6 = "Six";
420     string name7 = "Seven";
421     string name8 = "Eight";
422     string name9 = "Nine";
423     string name10 = "Ten";
424     string name11 = "Eleven";
425     string name12 = "Twelve";
426     string name13 = "Thirteen";
427     string name14 = "Fourteen";
428     string name15 = "Fifteen";
429     string name16 = "Sixteen";
430 
431     data.InsertStudent(1, name1);
```

```
432     data.InsertStudent(2, name2);
433     data.InsertStudent(3, name3);
434     data.InsertStudent(4, name4);
435     data.InsertStudent(5, name5);
436     data.InsertStudent(6, name6);
437     data.InsertStudent(7, name7);
438     data.InsertStudent(8, name8);
439     data.InsertStudent(9, name9);
440     data.InsertStudent(10, name10);
441     data.InsertStudent(11, name11);
442     data.InsertStudent(12, name12);
443     data.InsertStudent(13, name13);
444     data.InsertStudent(14, name14);
445     data.InsertStudent(15, name15);
446     data.InsertStudent(16, name16);
447 
448     data.Remove(12);
449 
450     std::vector<unsigned int> actualOutput = data.InOrder();
451     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14
    , 15, 16};
452     REQUIRE(expectedOutput.size() == actualOutput.size());
453     REQUIRE(actualOutput == expectedOutput);
454     REQUIRE(1 == 1);
455 }
456 
457 TEST_CASE("BST Remove Case 10", "[flag]"){
458 
459     GatorAVL data;
460 
461     string name1 = "One";
462     string name2 = "Two";
463     string name3 = "Three";
464     string name4 = "Four";
465     string name5 = "Five";
466     string name6 = "Six";
467     string name7 = "Seven";
468     string name8 = "Eight";
469     string name9 = "Nine";
470     string name10 = "Ten";
471     string name11 = "Eleven";
472     string name12 = "Twelve";
473     string name13 = "Thirteen";
474     string name14 = "Fourteen";
475     string name15 = "Fifteen";
476     string name16 = "Sixteen";
477 
478     data.InsertStudent(1, name1);
479     data.InsertStudent(2, name2);
480     data.InsertStudent(3, name3);
481     data.InsertStudent(4, name4);
482     data.InsertStudent(5, name5);
483     data.InsertStudent(6, name6);
484     data.InsertStudent(7, name7);
485     data.InsertStudent(8, name8);
```

```
486     data.InsertStudent(9, name9);
487     data.InsertStudent(10, name10);
488     data.InsertStudent(11, name11);
489     data.InsertStudent(12, name12);
490     data.InsertStudent(13, name13);
491     data.InsertStudent(14, name14);
492     data.InsertStudent(15, name15);
493     data.InsertStudent(16, name16);
494 
495     data.Remove(8);
496 
497     std::vector<unsigned int> actualOutput = data.InOrder();
498     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14
    , 15, 16};
499     REQUIRE(expectedOutput.size() == actualOutput.size());
500     REQUIRE(actualOutput == expectedOutput);
501     REQUIRE(1 == 1);
502 }
503 
504 TEST_CASE("Print Inorder", "[flag]") {
505 
506     GatorAVL data;
507 
508     string name1 = "One";
509     string name2 = "Two";
510     string name3 = "Three";
511     string name4 = "Four";
512     string name5 = "Five";
513     string name6 = "Six";
514     string name7 = "Seven";
515     string name8 = "Eight";
516     string name9 = "Nine";
517     string name10 = "Ten";
518     string name11 = "Eleven";
519     string name12 = "Twelve";
520     string name13 = "Thirteen";
521     string name14 = "Fourteen";
522     string name15 = "Fifteen";
523     string name16 = "Sixteen";
524 
525     data.InsertStudent(1, name1);
526     data.InsertStudent(2, name2);
527     data.InsertStudent(3, name3);
528     data.InsertStudent(4, name4);
529     data.InsertStudent(5, name5);
530     data.InsertStudent(6, name6);
531     data.InsertStudent(7, name7);
532 
533     ostringstream oss;
534     streambuf* p_cout_streambuf = std::cout.rdbuf();
535     cout.rdbuf(oss.rdbuf());
536 
537     data.PrintInOrder();
538 
539     cout.rdbuf(p_cout_streambuf); // restore
```

```
540
541     // test your oss content...
542     REQUIRE(oss.str() == "One, Two, Three, Four, Five, Six, Seven\n");
543 }
544
545 TEST_CASE("Print Preorder", "[flag]") {
546
547     GatorAVL data;
548
549     string name1 = "One";
550     string name2 = "Two";
551     string name3 = "Three";
552     string name4 = "Four";
553     string name5 = "Five";
554     string name6 = "Six";
555     string name7 = "Seven";
556     string name8 = "Eight";
557     string name9 = "Nine";
558     string name10 = "Ten";
559     string name11 = "Eleven";
560     string name12 = "Twelve";
561     string name13 = "Thirteen";
562     string name14 = "Fourteen";
563     string name15 = "Fifteen";
564     string name16 = "Sixteen";
565
566     data.InsertStudent(1, name1);
567     data.InsertStudent(2, name2);
568     data.InsertStudent(3, name3);
569     data.InsertStudent(4, name4);
570     data.InsertStudent(5, name5);
571     data.InsertStudent(6, name6);
572     data.InsertStudent(7, name7);
573
574     ostringstream oss;
575     streambuf *p_cout_streambuf = std::cout.rdbuf();
576     cout.rdbuf(oss.rdbuf());
577
578     data.PrintPreOrder();
579
580     cout.rdbuf(p_cout_streambuf); // restore
581
582     // test your oss content...
583     REQUIRE(oss.str() == "Four, Two, One, Three, Six, Five, Seven\n");
584 }
585
586 TEST_CASE("Print Postorder", "[flag]") {
587
588     GatorAVL data;
589
590     string name1 = "One";
591     string name2 = "Two";
592     string name3 = "Three";
593     string name4 = "Four";
594     string name5 = "Five";
```

```
595     string name6 = "Six";
596     string name7 = "Seven";
597     string name8 = "Eight";
598     string name9 = "Nine";
599     string name10 = "Ten";
600     string name11 = "Eleven";
601     string name12 = "Twelve";
602     string name13 = "Thirteen";
603     string name14 = "Fourteen";
604     string name15 = "Fifteen";
605     string name16 = "Sixteen";
606 
607     data.InsertStudent(1, name1);
608     data.InsertStudent(2, name2);
609     data.InsertStudent(3, name3);
610     data.InsertStudent(4, name4);
611     data.InsertStudent(5, name5);
612     data.InsertStudent(6, name6);
613     data.InsertStudent(7, name7);
614 
615     ostringstream oss;
616     streambuf *p_cout_streambuf = std::cout.rdbuf();
617     cout.rdbuf(oss.rdbuf());
618 
619     data.PrintPostOrder();
620 
621     cout.rdbuf(p_cout_streambuf); // restore
622 
623     // test your oss content...
624     REQUIRE(oss.str() == "One, Three, Two, Five, Seven, Six, Four\n");
625 }
626 
627 TEST_CASE("Search ID Success", "[flag]") {
628 
629 GatorAVL data;
630 
631 string name1 = "One";
632 string name2 = "Two";
633 string name3 = "Three";
634 string name4 = "Four";
635 string name5 = "Five";
636 string name6 = "Six";
637 string name7 = "Seven";
638 string name8 = "Eight";
639 string name9 = "Nine";
640 string name10 = "Ten";
641 string name11 = "Eleven";
642 string name12 = "Twelve";
643 string name13 = "Thirteen";
644 string name14 = "Fourteen";
645 string name15 = "Fifteen";
646 string name16 = "Sixteen";
647 
648 data.InsertStudent(1, name1);
649 data.InsertStudent(2, name2);
```

```
650 data.InsertStudent(3, name3);
651 data.InsertStudent(4, name4);
652 data.InsertStudent(5, name5);
653 data.InsertStudent(6, name6);
654 data.InsertStudent(7, name7);
655
656 ostringstream oss;
657 streambuf *p_cout_streambuf = std::cout.rdbuf();
658 cout.rdbuf(oss.rdbuf());
659
660 REQUIRE(data.SearchID(4) == 1);
661
662 cout.rdbuf(p_cout_streambuf); // restore
663
664 // test your oss content...
665 REQUIRE(oss.str() == "Four\n");
666 }
667
668 TEST_CASE("Search ID Fail", "[flag]") {
669
670     GatorAVL data;
671
672     string name1 = "One";
673     string name2 = "Two";
674     string name3 = "Three";
675     string name4 = "Four";
676     string name5 = "Five";
677     string name6 = "Six";
678     string name7 = "Seven";
679     string name8 = "Eight";
680     string name9 = "Nine";
681     string name10 = "Ten";
682     string name11 = "Eleven";
683     string name12 = "Twelve";
684     string name13 = "Thirteen";
685     string name14 = "Fourteen";
686     string name15 = "Fifteen";
687     string name16 = "Sixteen";
688
689     data.InsertStudent(1, name1);
690     data.InsertStudent(2, name2);
691     data.InsertStudent(3, name3);
692     data.InsertStudent(4, name4);
693     data.InsertStudent(5, name5);
694     data.InsertStudent(6, name6);
695     data.InsertStudent(7, name7);
696
697     ostringstream oss;
698     streambuf *p_cout_streambuf = std::cout.rdbuf();
699     cout.rdbuf(oss.rdbuf());
700
701     REQUIRE(data.SearchID(21) == 0);
702
703     cout.rdbuf(p_cout_streambuf); // restore
704
```

```
705 // test your oss content...
706     REQUIRE(oss.str() == "");
707 }
708
709 TEST_CASE("Search Name Success 1", "[flag]") {
710
711     GatorAVL data;
712
713     string name1 = "One";
714     string name2 = "Two";
715     string name3 = "Three";
716     string name4 = "Four";
717     string name5 = "Five";
718     string name6 = "Six";
719     string name7 = "Seven";
720     string name8 = "Eight";
721     string name9 = "Nine";
722     string name10 = "Ten";
723     string name11 = "Eleven";
724     string name12 = "Twelve";
725     string name13 = "Thirteen";
726     string name14 = "Fourteen";
727     string name15 = "Fifteen";
728     string name16 = "Sixteen";
729
730     data.InsertStudent(1, name1);
731     data.InsertStudent(2, name2);
732     data.InsertStudent(3, name3);
733     data.InsertStudent(4, name4);
734     data.InsertStudent(5, name5);
735     data.InsertStudent(6, name6);
736     data.InsertStudent(7, name7);
737
738     ostringstream oss;
739     streambuf *p_cout_streambuf = std::cout.rdbuf();
740     cout.rdbuf(oss.rdbuf());
741
742     REQUIRE(data.SearchName(name3) == 1);
743
744     cout.rdbuf(p_cout_streambuf); // restore
745
746 // test your oss content...
747     REQUIRE(oss.str() == "00000003\n");
748 }
749
750 TEST_CASE("Search Name Success 2", "[flag]") {
751
752     GatorAVL data;
753
754     string name1 = "One";
755     string name2 = "Two";
756     string name3 = "Three";
757     string name4 = "Four";
758     string name5 = "Five";
759     string name6 = "Six";
```

```
760     string name7 = "Seven";
761     string name8 = "Eight";
762     string name9 = "Nine";
763     string name10 = "Ten";
764     string name11 = "Eleven";
765     string name12 = "Twelve";
766     string name13 = "Thirteen";
767     string name14 = "Fourteen";
768     string name15 = "Fifteen";
769     string name16 = "Sixteen";
770 
771     data.InsertStudent(12345678, name1);
772     data.InsertStudent(23456781, name2);
773     data.InsertStudent(34567812, name3);
774     data.InsertStudent(45678123, name4);
775     data.InsertStudent(56781234, name5);
776     data.InsertStudent(67812345, name6);
777     data.InsertStudent(78123456, name7);
778 
779     ostringstream oss;
780     streambuf *p_cout_streambuf = std::cout.rdbuf();
781     cout.rdbuf(oss.rdbuf());
782 
783     REQUIRE(data.SearchName(name7) == 1);
784 
785     cout.rdbuf(p_cout_streambuf); // restore
786 
787 // test your oss content...
788     REQUIRE(oss.str() == "78123456\n");
789 }
790 
791 TEST_CASE("Search Name Fail", "[flag]") {
792 
793     GatorAVL data;
794 
795     string name1 = "One";
796     string name2 = "Two";
797     string name3 = "Three";
798     string name4 = "Four";
799     string name5 = "Five";
800     string name6 = "Six";
801     string name7 = "Seven";
802     string name8 = "Eight";
803     string name9 = "Nine";
804     string name10 = "Ten";
805     string name11 = "Eleven";
806     string name12 = "Twelve";
807     string name13 = "Thirteen";
808     string name14 = "Fourteen";
809     string name15 = "Fifteen";
810     string name16 = "Sixteen";
811 
812     data.InsertStudent(12345678, name1);
813     data.InsertStudent(23456781, name2);
814     data.InsertStudent(34567812, name3);
```

```
815     data.InsertStudent(45678123, name4);
816     data.InsertStudent(56781234, name5);
817     data.InsertStudent(67812345, name6);
818     data.InsertStudent(78123456, name7);
819 
820     ostringstream oss;
821     streambuf *p_cout_streambuf = std::cout.rdbuf();
822     cout.rdbuf(oss.rdbuf());
823 
824     REQUIRE(data.SearchName(name16) == 0);
825 
826     cout.rdbuf(p_cout_streambuf); // restore
827 
828 // test your oss content...
829     REQUIRE(oss.str() == "");
830 }
831 
832 TEST_CASE("Print Level Count - Empty", "[flag]") {
833 
834     GatorAVL data;
835 
836     ostringstream oss;
837     streambuf *p_cout_streambuf = std::cout.rdbuf();
838     cout.rdbuf(oss.rdbuf());
839 
840     data.PrintLevelCount();
841 
842     cout.rdbuf(p_cout_streambuf); // restore
843 
844 // test your oss content...
845     REQUIRE(oss.str() == "0\n");
846 }
847 
848 TEST_CASE("Print Level Count 1", "[flag]") {
849 
850     GatorAVL data;
851 
852     string name1 = "One";
853     string name2 = "Two";
854     string name3 = "Three";
855     string name4 = "Four";
856     string name5 = "Five";
857     string name6 = "Six";
858     string name7 = "Seven";
859     string name8 = "Eight";
860     string name9 = "Nine";
861     string name10 = "Ten";
862     string name11 = "Eleven";
863     string name12 = "Twelve";
864     string name13 = "Thirteen";
865     string name14 = "Fourteen";
866     string name15 = "Fifteen";
867     string name16 = "Sixteen";
868 
869     data.InsertStudent(12345678, name1);
```

```
870
871     ostringstream oss;
872     streambuf *p_cout_streambuf = std::cout.rdbuf();
873     cout.rdbuf(oss.rdbuf());
874
875     data.PrintLevelCount();
876
877     cout.rdbuf(p_cout_streambuf); // restore
878
879 // test your oss content...
880     REQUIRE(oss.str() == "1\n");
881 }
882
883 TEST_CASE("Print Level Count 5", "[flag]") {
884
885     GatorAVL data;
886
887     string name1 = "One";
888     string name2 = "Two";
889     string name3 = "Three";
890     string name4 = "Four";
891     string name5 = "Five";
892     string name6 = "Six";
893     string name7 = "Seven";
894     string name8 = "Eight";
895     string name9 = "Nine";
896     string name10 = "Ten";
897     string name11 = "Eleven";
898     string name12 = "Twelve";
899     string name13 = "Thirteen";
900     string name14 = "Fourteen";
901     string name15 = "Fifteen";
902     string name16 = "Sixteen";
903
904     data.InsertStudent(1, name1);
905     data.InsertStudent(2, name2);
906     data.InsertStudent(3, name3);
907     data.InsertStudent(4, name4);
908     data.InsertStudent(5, name5);
909     data.InsertStudent(6, name6);
910     data.InsertStudent(7, name7);
911     data.InsertStudent(8, name8);
912     data.InsertStudent(9, name9);
913     data.InsertStudent(10, name10);
914     data.InsertStudent(11, name11);
915     data.InsertStudent(12, name12);
916     data.InsertStudent(13, name13);
917     data.InsertStudent(14, name14);
918     data.InsertStudent(15, name15);
919     data.InsertStudent(16, name16);
920
921     ostringstream oss;
922     streambuf *p_cout_streambuf = std::cout.rdbuf();
923     cout.rdbuf(oss.rdbuf());
924
```

```
925     data.PrintLevelCount();
926 
927     cout.rdbuf(p_cout_streambuf); // restore
928 
929 // test your oss content...
930     REQUIRE(oss.str() == "5\n");
931 }
932 
933 TEST_CASE("Print Level Count 4", "[flag]") {
934 
935     GatorAVL data;
936 
937     string name1 = "One";
938     string name2 = "Two";
939     string name3 = "Three";
940     string name4 = "Four";
941     string name5 = "Five";
942     string name6 = "Six";
943     string name7 = "Seven";
944     string name8 = "Eight";
945     string name9 = "Nine";
946     string name10 = "Ten";
947     string name11 = "Eleven";
948     string name12 = "Twelve";
949     string name13 = "Thirteen";
950     string name14 = "Fourteen";
951     string name15 = "Fifteen";
952     string name16 = "Sixteen";
953 
954     data.InsertStudent(1, name1);
955     data.InsertStudent(2, name2);
956     data.InsertStudent(3, name3);
957     data.InsertStudent(4, name4);
958     data.InsertStudent(5, name5);
959     data.InsertStudent(6, name6);
960     data.InsertStudent(7, name7);
961     data.InsertStudent(8, name8);
962     data.InsertStudent(9, name9);
963     data.InsertStudent(10, name10);
964     data.InsertStudent(11, name11);
965     data.InsertStudent(12, name12);
966     data.InsertStudent(13, name13);
967     data.InsertStudent(14, name14);
968     data.InsertStudent(15, name15);
969     //data.InsertStudent(16, name16);
970 
971     ostringstream oss;
972     streambuf *p_cout_streambuf = std::cout.rdbuf();
973     cout.rdbuf(oss.rdbuf());
974 
975     data.PrintLevelCount();
976 
977     cout.rdbuf(p_cout_streambuf); // restore
978 
979 // test your oss content...
```

```
 980     REQUIRE(oss.str() == "4\n");
 981 }
 982
 983 TEST_CASE("Remove Inorder - First Success", "[flag]") {
 984
 985     GatorAVL data;
 986
 987     string name1 = "One";
 988     string name2 = "Two";
 989     string name3 = "Three";
 990     string name4 = "Four";
 991     string name5 = "Five";
 992     string name6 = "Six";
 993     string name7 = "Seven";
 994     string name8 = "Eight";
 995     string name9 = "Nine";
 996     string name10 = "Ten";
 997     string name11 = "Eleven";
 998     string name12 = "Twelve";
 999     string name13 = "Thirteen";
1000     string name14 = "Fourteen";
1001     string name15 = "Fifteen";
1002     string name16 = "Sixteen";
1003
1004     data.InsertStudent(1, name1);
1005     data.InsertStudent(2, name2);
1006     data.InsertStudent(3, name3);
1007     data.InsertStudent(4, name4);
1008     data.InsertStudent(5, name5);
1009     data.InsertStudent(6, name6);
1010     data.InsertStudent(7, name7);
1011     data.InsertStudent(8, name8);
1012     data.InsertStudent(9, name9);
1013     data.InsertStudent(10, name10);
1014     data.InsertStudent(11, name11);
1015     data.InsertStudent(12, name12);
1016     data.InsertStudent(13, name13);
1017     data.InsertStudent(14, name14);
1018     data.InsertStudent(15, name15);
1019     data.InsertStudent(16, name16);
1020
1021     REQUIRE(data.RemoveInOrder(0) == 1);
1022
1023     std::vector<unsigned int> actualOutput = data.InOrder();
1024     std::vector<unsigned int> expectedOutput = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
     14, 15, 16};
1025     REQUIRE(expectedOutput.size() == actualOutput.size());
1026     REQUIRE(actualOutput == expectedOutput);
1027     REQUIRE(1 == 1);
1028 }
1029
1030 TEST_CASE("Remove Inorder - Last Success", "[flag]") {
1031
1032     GatorAVL data;
1033
```

```
1034     string name1 = "One";
1035     string name2 = "Two";
1036     string name3 = "Three";
1037     string name4 = "Four";
1038     string name5 = "Five";
1039     string name6 = "Six";
1040     string name7 = "Seven";
1041     string name8 = "Eight";
1042     string name9 = "Nine";
1043     string name10 = "Ten";
1044     string name11 = "Eleven";
1045     string name12 = "Twelve";
1046     string name13 = "Thirteen";
1047     string name14 = "Fourteen";
1048     string name15 = "Fifteen";
1049     string name16 = "Sixteen";
1050 
1051     data.InsertStudent(1, name1);
1052     data.InsertStudent(2, name2);
1053     data.InsertStudent(3, name3);
1054     data.InsertStudent(4, name4);
1055     data.InsertStudent(5, name5);
1056     data.InsertStudent(6, name6);
1057     data.InsertStudent(7, name7);
1058     data.InsertStudent(8, name8);
1059     data.InsertStudent(9, name9);
1060     data.InsertStudent(10, name10);
1061     data.InsertStudent(11, name11);
1062     data.InsertStudent(12, name12);
1063     data.InsertStudent(13, name13);
1064     data.InsertStudent(14, name14);
1065     data.InsertStudent(15, name15);
1066     data.InsertStudent(16, name16);
1067 
1068     REQUIRE(data.RemoveInOrder(15) == 1);
1069 
1070     std::vector<unsigned int> actualOutput = data.InOrder();
1071     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
     , 14, 15};
1072     REQUIRE(expectedOutput.size() == actualOutput.size());
1073     REQUIRE(actualOutput == expectedOutput);
1074     REQUIRE(1 == 1);
1075 }
1076 
1077 TEST_CASE("Remove Inorder - Fail", "[flag]") {
1078 
1079     GatorAVL data;
1080 
1081     string name1 = "One";
1082     string name2 = "Two";
1083     string name3 = "Three";
1084     string name4 = "Four";
1085     string name5 = "Five";
1086     string name6 = "Six";
1087     string name7 = "Seven";
```

```
1088     string name8 = "Eight";
1089     string name9 = "Nine";
1090     string name10 = "Ten";
1091     string name11 = "Eleven";
1092     string name12 = "Twelve";
1093     string name13 = "Thirteen";
1094     string name14 = "Fourteen";
1095     string name15 = "Fifteen";
1096     string name16 = "Sixteen";
1097 
1098     data.InsertStudent(1, name1);
1099     data.InsertStudent(2, name2);
1100     data.InsertStudent(3, name3);
1101     data.InsertStudent(4, name4);
1102     data.InsertStudent(5, name5);
1103     data.InsertStudent(6, name6);
1104     data.InsertStudent(7, name7);
1105     data.InsertStudent(8, name8);
1106     data.InsertStudent(9, name9);
1107     data.InsertStudent(10, name10);
1108     data.InsertStudent(11, name11);
1109     data.InsertStudent(12, name12);
1110     data.InsertStudent(13, name13);
1111     data.InsertStudent(14, name14);
1112     data.InsertStudent(15, name15);
1113     data.InsertStudent(16, name16);
1114 
1115     REQUIRE(data.RemoveInOrder(16) == 0);
1116 
1117     std::vector<unsigned int> actualOutput = data.InOrder();
1118     std::vector<unsigned int> expectedOutput = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
     , 14, 15, 16};
1119     REQUIRE(expectedOutput.size() == actualOutput.size());
1120     REQUIRE(actualOutput == expectedOutput);
1121     REQUIRE(1 == 1);
1122 }
1123 
1124 TEST_CASE("Insert Remove Insert", "[flag]") {
1125 
1126     GatorAVL data;
1127 
1128     string name1 = "One";
1129     string name2 = "Two";
1130     string name3 = "Three";
1131     string name4 = "Four";
1132     string name5 = "Five";
1133     string name6 = "Six";
1134     string name7 = "Seven";
1135     string name8 = "Eight";
1136     string name9 = "Nine";
1137     string name10 = "Ten";
1138     string name11 = "Eleven";
1139     string name12 = "Twelve";
1140     string name13 = "Thirteen";
1141     string name14 = "Fourteen";
```

```
1142     string name15 = "Fifteen";
1143     string name16 = "Sixteen";
1144 
1145     data.InsertStudent(1, name1);
1146     data.InsertStudent(2, name2);
1147     data.InsertStudent(3, name3);
1148 
1149     std::vector<unsigned int> actualOutput = data.InOrder();
1150     std::vector<unsigned int> expectedOutput = {1, 2, 3};
1151     REQUIRE(expectedOutput.size() == actualOutput.size());
1152     REQUIRE(actualOutput == expectedOutput);
1153     REQUIRE(1 == 1);
1154 
1155 
1156     data.Remove(3);
1157 
1158     actualOutput = data.InOrder();
1159     expectedOutput = {1, 2};
1160     REQUIRE(expectedOutput.size() == actualOutput.size());
1161     REQUIRE(actualOutput == expectedOutput);
1162     REQUIRE(1 == 1);
1163 
1164     data.InsertStudent(4, name4);
1165 
1166     actualOutput = data.InOrder();
1167     expectedOutput = {1, 2, 4};
1168     REQUIRE(expectedOutput.size() == actualOutput.size());
1169     REQUIRE(actualOutput == expectedOutput);
1170     REQUIRE(1 == 1);
1171 }
1172 
1173 TEST_CASE("Large Insert", "[flag]"){
1174 
1175     GatorAVL tree;   // Create a Tree object
1176     string name = "Bob";
1177 
1178     for(unsigned int i = 0; i < 10000; i++) {
1179         tree.InsertStudent(i, name);
1180     }
1181     vector<unsigned int> expectedOutput(10000);
1182     for(unsigned int i = 0; i < 10000; i++) {
1183         expectedOutput[i] = i;
1184     }
1185 
1186     std::vector<unsigned int> actualOutput = tree.InOrder();
1187     REQUIRE(expectedOutput.size() == actualOutput.size());
1188     REQUIRE(actualOutput == expectedOutput);
1189 
1190     REQUIRE(1 == 1);
1191 }
1192 
1193 
```