

## מטלה 4 – איטרטורים

מגישה 211548268

### הרצת התוכנית

כדי להריץ את התוכנית כותבים `make tree` בעזרת הרצה אוטומטית של `./tree_app`.  
כדי להריץ את הטסטים כותבים `make tests` בעזרת הרצה אוטומטית של `./tree_tests`.

### מחלקות

#### tree.hpp

המחלקה העיקרית בה נמצאים המבנים Node, tree, TreeGui. צומת בעץ שיכול לאכסן בתוכו כל ערך (template) ווקטור של בנים. לצומת יש שני בנאים, הדיפולטיבי שלתוכו מכניסים רק ערך אחד יוצר צומת שיכולים להיות לו עד שני בנים ואילו הבנאי השני מקבל שני ערכים שאחד מהם זה מספר הילדים המקסימלי שיכול להיות לו.

tree- העץ. מקשר בין כל הצמתים. העץ מקבל את השורש ובעל מטודות להוספת בנים. לעץ יש כמה תתי מחלקות שהן האיטרטורים: preorder, inorder, postorder, dfs, bfs, minheap. כאשר העץ הוא בינארי מתבצעת סריקה רגילה על פי הכלל המקובל. במקרה אחר מתבצעת סריקת dfs preorder, inorder, postorder.

TreeGui- המחלקה הגרפית שמייצגת את העץ. הגרפיקה נעשית דרך ספריית SFML. הגרפיקה רצה כאשר יוצרים אובייקט מסוג TreeGui ומפעילים את המטודה run.

#### Preorder.cpp, Inorder.cpp, Postorder.cpp, DFS.cpp, BFS.cpp, Heap.cpp

הקבצים שבהם מיושמים האיטרטורים. דרכי הפעולה מפורטות בדוקומנטציה בקוד.

#### Complex.hpp

מחלקה המייצגת מספר מרוכב.

#### test\_xxxx.cpp

לכל מחלקה בתוכנית קיים טסט. להלן הדגמה של הרצת הטסטים:

```
liza@liza-VirtualBox: ~/workcpp/Assignment4
RUN      BFSTest.KaryTreeTraversal
OK       BFSTest.KaryTreeTraversal (0 ms)
RUN      BFSTest.KaryTreeTraversal_Chars
OK       BFSTest.KaryTreeTraversal_Chars (0 ms)
RUN      BFSTest.BinaryTreeTraversal
OK       BFSTest.BinaryTreeTraversal (0 ms)
RUN      BFSTest.BinaryTreeTraversalComplex
OK       BFSTest.BinaryTreeTraversalComplex (0 ms)
----- 4 tests from BFSTest (0 ms total)

----- 1 test from MinHeapIteratorTest
RUN      MinHeapIteratorTest.DereferenceNullPointer
OK       MinHeapIteratorTest.DereferenceNullPointer (0 ms)
----- 1 test from MinHeapIteratorTest (0 ms total)

----- 2 tests from MinHeapTest
RUN      MinHeapTest.KaryTreeTraversal
OK       MinHeapTest.KaryTreeTraversal (0 ms)
RUN      MinHeapTest.BinaryTreeTraversal
OK       MinHeapTest.BinaryTreeTraversal (0 ms)
----- 2 tests from MinHeapTest (0 ms total)

----- 7 tests from ComplexTest
RUN      ComplexTest.DefaultConstructor
OK       ComplexTest.DefaultConstructor (0 ms)
RUN      ComplexTest.ParameterizedConstructor
OK       ComplexTest.ParameterizedConstructor (0 ms)
RUN      ComplexTest.Magnitude
OK       ComplexTest.Magnitude (0 ms)
RUN      ComplexTest.ComparisonOperators
OK       ComplexTest.ComparisonOperators (0 ms)
RUN      ComplexTest.LessThanOrEqual
OK       ComplexTest.LessThanOrEqual (0 ms)
RUN      ComplexTest.GreaterThanOrEqual
OK       ComplexTest.GreaterThanOrEqual (0 ms)
RUN      ComplexTest.OutputStreamOperator
OK       ComplexTest.OutputStreamOperator (0 ms)
----- 7 tests from ComplexTest (0 ms total)

----- Global test environment tear-down
===== 35 tests from 17 test suites ran. (1 ms total)
PASSED   35 tests.

liza@liza-VirtualBox: ~/workcpp/Assignment4$

liza@liza-VirtualBox: ~/workcpp/Assignment4$ make tests
g++ -std=c++17 -Wall -g -c test_complex.cpp -o test_complex.o
g++ -std=c++17 -Wall -g -o tree_tests test_main.o test_preorder.o test_inorder.o
test_tree.o test_dfs.o test_bfs.o test_heap.o test_complex.o PreOrder.o PostOrder.o InOrder.o DF
S.o BFS.o Heap.o -lgtest -lgtest_main -lpthread
./tree_tests
===== Running 35 tests from 17 test suites.
----- Global test environment set-up.
----- 1 test from PreOrderIteratorTest
RUN      PreOrderIteratorTest.DereferenceNullPointer
OK       PreOrderIteratorTest.DereferenceNullPointer (0 ms)
----- 1 test from PreOrderIteratorTest (0 ms total)

----- 1 test from PREORDERtest_3_ary_tree
RUN      PREORDERtest_3_ary_tree.tst1
OK       PREORDERtest_3_ary_tree.tst1 (0 ms)
----- 1 test from PREORDERtest_3_ary_tree (0 ms total)

----- 2 tests from PREORDERtest_binary_tree
RUN      PREORDERtest_binary_tree.tst2
OK       PREORDERtest_binary_tree.tst2 (0 ms)
RUN      PREORDERtest_binary_tree.tst2Complex
OK       PREORDERtest_binary_tree.tst2Complex (0 ms)
----- 2 tests from PREORDERtest_binary_tree (0 ms total)

----- 1 test from PostOrderIteratorTest
RUN      PostOrderIteratorTest.DereferenceNullPointer
OK       PostOrderIteratorTest.DereferenceNullPointer (0 ms)
----- 1 test from PostOrderIteratorTest (0 ms total)

----- 1 test from POSTORDERtest_3_ary_tree
RUN      POSTORDERtest_3_ary_tree.tst1
OK       POSTORDERtest_3_ary_tree.tst1 (0 ms)
----- 1 test from POSTORDERtest_3_ary_tree (0 ms total)

----- 2 tests from POSTORDERtest_binary_tree
RUN      POSTORDERtest_binary_tree.tst2
OK       POSTORDERtest_binary_tree.tst2 (0 ms)
RUN      POSTORDERtest_binary_tree.tst2Complex
OK       POSTORDERtest_binary_tree.tst2Complex (0 ms)
----- 2 tests from POSTORDERtest_binary_tree (0 ms total)

----- 1 test from InOrderIteratorTest
```

## Main.cpp

מדגים את אופן פעולת התוכנית.

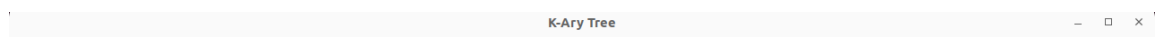
בזמן הרצה נקבל הדפסה ונפתח חלון עם ציור של העץ, ההדפסה נפסקת כל עוד החלון פתוח. לאחר סגירת החלון צץ חלון נוסף וההדפסה ממשיכה. סך הכל אמורים להפתח שלושה חלונות עם ציורים של עצים שונים.

העץ הראשון יכול להיות עם 3 בנים לכל צומת והוא מחזיק מספרים מרוכבים.

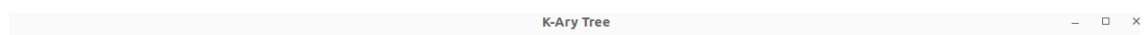
העץ השני הוא עץ בינארי של אותיות.

העץ השלישי הוא עץ בינארי של מספרים מרוכבים.

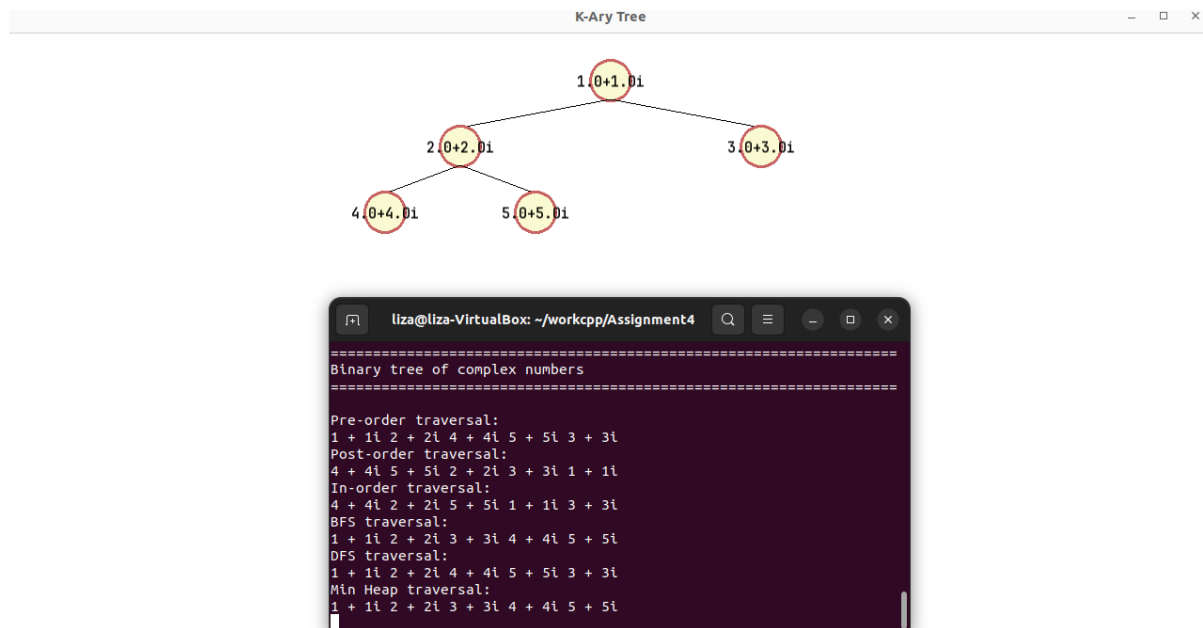
להלן דוגמאות של איך שזה אמור להראות:



```
liza@liza-VirtualBox: ~/workcpp/Assignment4
liza@liza-VirtualBox:~/workcpp/Assignment4$ make tree
./tree_app
=====
3-ary tree of complex numbers
=====
Pre-order traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
Post-order traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
In-order traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
BFS traversal:
7 + 7i 0 + 0i 2 + 2i 3 + 3i 4 + 4i 1 + 1i 5 + 5i 6 + 6i 8 + 8i
DFS traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
Min Heap traversal:
0 + 0i 1 + 1i 2 + 2i 3 + 3i 4 + 4i 5 + 5i 6 + 6i 7 + 7i 8 + 8i
```



```
liza@liza-VirtualBox: ~/workcpp/Assignment4
=====
Binary tree of chars
=====
Pre-order traversal:
A B D E C
Post-order traversal:
D E B C A
In-order traversal:
D B E A C
BFS traversal:
A B C D E
DFS traversal:
A B D E C
Heap traversal:
A B C D E
```



## ספריות

מבני נתונים שיש בהם שימוש בתוכנית:

- queue
- vector
- stack

לחישוב מרחק בין מספרים מרוכבים (שימוש בשורש מתוך הספרייה):

- cmath

לדיוק של מספר אחד אחרי הנקודה בהחזרת מחזרת:

- iomanip

המחלקה הגפרית שהשתמשתי בה לGUI:

- SMFL/Graphics.hpp

לזרקת שגיאות:

- stdexcept

להדפסות ולבדיקת הדפסה של myHeap:

- iostream

למטודות to\_string:

- string

לטסטים:

- gtest/gtest.h

## הערת valgrind

שמתי לב שהספרייה הגרפית גורמת לזליגת זכרון קלה של 56 בתים. לא מדובר באובייקט TreeGui כיוון שאין שימוש בהקצאת זכרון לheap (new), מקריאה בפורומים הבנתי שלספרייה יש משתנים גלובליים שלא מתנקים עד שהתוכנית מסתיימת והבדיקה קורת לפני שההקצאות הנ"ל מתנקות. להלן דוגמה של בדיקת זליגות זכרון תקינה כאשר סימנתי את האובייקט של gui בהערה:

```
liza@liza-VirtualBox:~/workcpp/Assignment4$ valgrind ./tree_app
==31329== Memcheck, a memory error detector
==31329== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==31329== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==31329== Command: ./tree_app
==31329==
=====
3-ary tree of complex numbers
=====

Pre-order traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
Post-order traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
In-order traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
BFS traversal:
7 + 7i 0 + 0i 2 + 2i 3 + 3i 4 + 4i 1 + 1i 5 + 5i 6 + 6i 8 + 8i
DFS traversal:
7 + 7i 0 + 0i 2 + 2i 4 + 4i 1 + 1i 3 + 3i 5 + 5i 6 + 6i 8 + 8i
Min Heap traversal:
0 + 0i 1 + 1i 2 + 2i 3 + 3i 4 + 4i 5 + 5i 6 + 6i 7 + 7i 8 + 8i

=====
Binary tree of chars
=====

Pre-order traversal:
A B D E C
Post-order traversal:
D E B C A
In-order traversal:
D B E A C
BFS traversal:
A B C D E
DFS traversal:
A B D E C
Heap traversal:
A B C D E

=====
Binary tree of complex numbers
=====

Pre-order traversal:
1 + 1i 2 + 2i 4 + 4i 5 + 5i 3 + 3i
Post-order traversal:
4 + 4i 5 + 5i 2 + 2i 3 + 3i 1 + 1i
In-order traversal:
4 + 4i 2 + 2i 5 + 5i 1 + 1i 3 + 3i
BFS traversal:
1 + 1i 2 + 2i 3 + 3i 4 + 4i 5 + 5i
DFS traversal:
1 + 1i 2 + 2i 4 + 4i 5 + 5i 3 + 3i
Min Heap traversal:
1 + 1i 2 + 2i 3 + 3i 4 + 4i 5 + 5i
==31329==
==31329== HEAP SUMMARY:
==31329==    in use at exit: 0 bytes in 0 blocks
==31329==   total heap usage: 491 allocs, 491 frees, 195,624 bytes allocated
==31329==
==31329== All heap blocks were freed -- no leaks are possible
==31329==
==31329== For lists of detected and suppressed errors, rerun with: -s
==31329== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
liza@liza-VirtualBox:~/workcpp/Assignment4$
```