**Node**: מכיל את המידע בvalue- מכיל מצביעים אל ההורה שלו, האחים שלו ולילד הראשון והאחרון שלו(כך שאפשר להגיע מילד ראשון/אחרון אל שאר הילדים אם לא בינארי) ובנוסף מכיל את מספר הילדים. הוא ניתן להשוואה אם הערך שלו ניתן להשוואה. הוא מה שנכנס לעץ.

‎**Tree**

מחלקה של עץ השומרת את השורש שלה ואת מספר הילדים המקסימלי של-נוד(NODE) יכול להיות.

מכיל את הפונקציות:

add\_root(Node<T> &n): מכניסה את הנוד שניתן להיות השורש.

add\_sub\_node(Node<T> &node, Node<T> &child): במידה ומספר הילדים ב-"node" הוא נמוך מהגבול מכניס את "child" להיות בן ומשנה את ערכי המצביעים שלו אל אחיו והורו להיות מה שצריך.

מכיל איטרטורים ופונקציות מתאימות כוללות begin ,end לגישה אל האיטרטורים:

ForNonBinaryTreesIterator: זהו מחלקת תוסף לאיטרטורים שעובדים על רק עץ בינארי, שנותן לוגיקת איטרטור אחרת (DFS) כאשר קוראים לאיטרטורים לעצים בינאריים כאשר העצים אינם בינארים.

BFSIterator: איטרטור שמתחיל מroot של עץ ועובר על האיברים כBFS.

DFSIterator: איטרטור שמתחיל מroot של עץ ועובר על האיברים DFS.

InOrderIterator, PreOrderIterator, PostOrderIterator: איטרטורים לעץ בינארי. כל אחד מהם סורק את העץ בסדר שרשום במחלקה (InOrderIterator-in order scan).

MinHeapIterator: איטרטור לעץ בינארי שסורק את העץ בסדר עולה של הערכים.