Karl Goeltner
Joël Porquet
ECS 36C
05/07/2020

Programming #3 – Analysis Report

P1 – Multiset: My program multiset.h implements a multiset using a binary search tree data structure. Its public API includes Size, Empty, Insert, Remove, Contains, Count, Floor, Ceil, Max & Min. The helper methods of Insert, Remove, Search, Floor, Ceil, EmptyError, and Min aid in their implementation.

My node structure includes the standard key, left & right pointers, along with a quantity variable to keep track of duplicates. My public API methods are as follows: Size() & Empty() return values based on multiset size. Insert() & Remove() use their respective helpers to perform standard BST insertion/removal but also update the quantity variable for duplicates. Remove() also uses the Min() recursive method to deal the with the two children removal case. Contains() & Count() call helper Search() which iteratively looks for a specified node with its quantity in the multiset. Floor() & Ceil() call respective helpers and EmptyError() to ensure that the multiset isn't empty. They then recursively check if the node is NIL, equals the key, key is larger/smaller, before finally comparing with the opposite branch closest floor/ceil value and returning the node and key. Max() is an iterative method that traverses right-most. Min() calls its recursive helper that traverses left-most. Methods Remove()/Count()/Floor()/Ceil() throw exception "out_of_range" for "Invalid key" if the key is not within the multiset, and Floor()/Ceil()/Max()/Min() throw exception "underflow_error" for "Empty multiset" when a user tries to access an element of an empty multiset.

In test_multiset.cc, I created several tests to increase the coverage of my implementation. Besides the given tests Empty & Onekey which check an empty multiset and three insertions, FloorCeil & FloorCeilRemoval perform insertions and removals before checking floor & ceil of multiple values. CheckContains & RemoveContains check the containment of values after insertions and removals respectively. MaxMin checks maximum & minimum values after insertions & removals. ErrorCheck tests all methods that may throw exceptions. MultipleCount & RandomCount check Count, Contains, Max, Min after insertions & removals. OrderedDuplicates checks Floor, Ceil, Count, Contains, Max, Min after ordered insertions of duplicate values.

P2 – Prime Factors: My program prime_factors.cc performs a trivial prime factorization on a number and prints output based on the specified operation: all, max, min, near. It uses the prior implementation of multiset.

The logic of prime_factors.cc starts at the main method before flowing to respective methods which handle the various operations. The main method first performs error checking for proper number of arguments, a valid number, and if the number is 0 or 1 as they are special cases. It then calls the respective printing methods: printAll, printMax, printMin, printNear. They each call primeFactors() which stores the prime factors of the inputted number into multiset by first dealing with 2s for even case, odd primes, and the remaining value. printAll() prints all prime factors by obtaining the minimum value and iterating by attaining the next ceil. printMax() and printMin() print the maximum and minimum value respectively. printNear() deals with four cases of a '+' sign, '-' sign, prime factor itself, or no match of prime factors; checkNear() is initially called to check for a valid prime factor, and printNoMatch() is called when the specified prime factor is not found.

Sources:

Ferb. "Floor in Binary Search Tree (BST)." *GeeksforGeeks*, 16 May 2019, www.geeksforgeeks.org/floor-in-binary-search-tree-bst/.

Garg, Vishwas. "Efficient Program to Print All Prime Factors of a given Number." *GeeksforGeeks*, 20 Aug. 2019, www.geeksforgeeks.org/print-all-prime-factors-of-a-given-number/.