James Ng Homework 1 Question 2 Group 7

Objective is to search for a number in an array that is randomly generated for both 1 million and ten million numbers. First, I worked on Iterative Search and used the code from Blackboard in order to search throughout the array. I did the same for the recursive and linked type. For the all the searches, I created an array for both million and ten million. I actually tried creating a traditional array but for ten million , int array[10000000] caused me an error so I had to look up some solution and ended up creating a pointer for the array. Unfortunately, I needed to make a whole separate variable since “thesizeof(arr)/sizeof(*arr)” wouldn’t detect the whole size. After I created the array and filled it with random numbers up to one million, I sorted using the QuickSort code that was available on Blackboard. I decided to use QuickSort since it recursively sorted each element in the array and was the quickest way to complete the program. Afterwards, I prompted the user to give value for a number to see if that number and if the binary search didn’t find the number, it would tell the user there is no number within the array. After the user gives the value, the timer would start to time how fast each search was. I used both code from Blackboard, Stack Overflow and YouTube to make sure it recorded in milliseconds and how to start timing efficiently. After the timing ended, it would state how fast each search was. At first, I went with seconds for timing the search and I would get 0 seconds. So far, the most effective way to see how fast search was, I switched the time from milliseconds to seconds to see how fast more accurately it was. For my results, it seems Iterative Binary Search was the fastest out of all the searches. Though the time varied, it was the fastest out of them all. As for linked-type binary, I believe I was supposed a linked list. So, I used the code from Udemy to make the list. The program seems to be getting too large, so I decided to make a header file for the linked list. The linked list had both lists for one million and ten million, which also included a search function. However, I do not get a correct response as it keeps stating that it could not find the number within the linked list. For the other searches, I repeated the steps above.

James Ng Homework 1 Question 5 Group 7

Objective is to create a multi-level sort for both if n is greater than ten and n is less than ten. First, I used the same code from last question in order to time and Quicksort so I can sort and time the whole program. For SortY, I used the array and the reasonably comparable sizes to bring the data to sort using QuickSort. The for loop I used actually counts down from the top of the array, so I am able to get the bottom half of the array. Before the for loop starts, I start the timer so I can see how fast it sorts. I did a similar process with SortX, however, when the array was less than ten, I would call SortY. I had to include the break statement so it wouldn't keep sorting even after the function was called. For some reason if I made the size subtracted by i equal to 11, it would work fine. However, when I tried i > 10, it would not work as intended. As for the arrays, I made the sizes constant in any case they would be changed throughout the program. I made them similar in size so we could time the sorts. Not surprisingly SortY is similar in size due to them dealing with both ten numbers. Meanwhile, SortX is noticeable in time as the one with more numbers would obviously take longer to sort. For the array, I randomly generated numbers within one hundred and used a pointer to make the program run smooth.

James Ng Homework 1 Question 8 Group 7

The objective is to find all of the sum combinations of the value the user inputs. I created the class called Sum so it would point out that it is meant to find the sum. So, I made a function that would ask both size of the array and the sum that the array needed to find. For size, I made it so it would catch all the issues if the user would have input any negative number or 0 since that would not have done anything. Since we made a vector, I randomly generated numbers in the vector so we would have new numbers every single time we run the program. I used up to 100 for the random numbers to keep it simple. Afterwards, we had to find a way to link all the data to the class, so I made a class variable and used that to call the function to find combinations. In the public class, we create the function FindEveryCombination, which is made to find every single sum combination out of the whole array. The function is similar to Udemy's code from C++ Review video 63 with Combinations. I created a for loop that would find every possible way to add up to the user's inputted sum. Then I made a variable that would add up all the numbers into one value to see if they would add up to the sum. If we found a match, we would store that in another vector, since we do not know how many combinations there would be. The function recursively repeated until we have exhausted all possible combinations. In the public class, we created an overall vector that would store all the answers that we got in StoreEveryCombination. I also added a sort function from #algorithm to make it cleaner. I attempted to use QuickSort, but it became a mess to sort using that function. After the functions are done, I tidied up the listing of all numbers in the vector in order to make it look appealing, adding plus signs and brackets so it is cleaner. I created a variable that uses auto&, which accepts any value without question. I looked up on Stack Overflow what auto does from the timing code and turns out it can accept values from vectors. So, I used auto to link the data back to the main function and listed all the answers we have gotten from our class.

James Ng Homework 1 Question 11

Objective is to read the menu using derived classes, pure virtual function, and a templated class. First, we start off with main, which prompts the user to asking what restaurant they would like to eat at. I looked up on Stack Overflow to see if we can make capital letters not mean much for the input, which led to transform. I have tried to use a while loop to prevent the user from putting in an invalid statement, but it always looped so I had to get rid of that idea and stuck with the if statements. At first, I was completely stuck on why all the inputs were to go off at once. Originally, I had “if(name == “italian” ||”1” “, which would trigger the rest of the if statements. I was not aware of how the or statement completely worked so it provided a wall for a while. In those if statements, we linked each restaurant as they are all in public classes and used the input of the user to have Reader_Robot direct us to the correct menu. Reader_Robot is a templated class that uses any input to direct you to the restaurant you want. We have a class called Restaurant and I derived the specific restaurants to their respective names. I went through real restaurants and their menus so I could provide accurate data on each food item. I used vectors since I am not aware of the size and it is more flexible to use than array in this scenario.