Gretel Rajamoney

3/8/2020

Design Document #5

Understanding the Problem:

For this assignment I will be creating a program which implements a linked list using pointers and object-oriented programming. I must create this program from scratch without utilizing the offered implementations from the standard template library. The standard template libraries which are not allowed to implement within our program are the <list> and <forward_list> libraries. The linked list much be designed to contain signed integers of the data type 'int'. I am required to implement the classes and member functions which are already created on the Program 5 file on canvas. The classes which have been made for me are the Node class and the Linked_List class, within these classes are the member functions. The sort_ascending() function must be implemented using the recursive merge sort algorithm, while the sort_descending() function can be created utilizing the merge sort or using a different algorithm which is hinted in the extra credit section of the assignment document. Within a Linked_List we much also implement a function that counts the number of prime numbers that are present. This can either be within the Linked_List class or in a different class. Assume that negative numbers are never considered to be prime numbers. All program functionality must be implemented using the linked list and can not be stored using arrays, vectors, or other data structures. Assume that when sorting nodes you must not swap them. The user will be prompted to enter a number and then re-prompted over and over again until they say that they don't want anymore numbers. The numbers inputted by the user should be displayed in either ascending or descending order, with the total count of prime number that are in the list. Lastly, the program should prompt the user asking whether they want to repeat the entire process again.

Design:

1. Prompts user asking for a number
2. Asks whether user would like to enter another number
3. If yes, prompt user asking for the number
4. Continue steps 2 and 3 until the user says no
5. If no, prompt user asking whether they would like the list sorted in ascending or descending order
6. Sort the list
7. Calculate how many prime numbers are in the list

```
class Node {
public:
    int val;  //variable that holds the variable that the node stores
    Node *next;  //pointer that points to the following node in the list
//add constructors and other functionalities necessary for the program
}

class Linked_List {
private:
    unsigned int length;
    Node *head;
    //any other necessary functionalities for the program
Public:
    int get_length();
    //there is no set_length(unsigned int) because the reasoning should be intuitive)
    void print();  //outputs all of the integers in the list
    void clear();  //deletes the entire list and removes all the nodes and resets length to null
    unsigned int push_front(int);  //inserts a new value to the back of the list (returns new list length)
    unsigned int insert(int val, unsigned int index);  //inserts a new values into the list at a specified location (returns the new list length)
```

```
        void sort_ascending();  //sorts the nodes in the list in ascending order (implement
    merge sort algorithm)

        void sort_descending();  //sorts the nodes in the list in descending order (implement
    merge sort of algorithms stated in the extra credit)

        //extra member variables or functions necessary for the program

}

void selection_sort(int *arr, int size) {

        int i;

        int j;

        int min_index;

        //move boundary one by one

        for(i = 0; i < size; i++) {

                min_index = i;

        for(j = i + 1; j < size; j++) {

                if(*(arr + j) <*(arr + min_index)) min_index = j;

        }

        //swaps the minimum element with the current first element

        int temp;

        temp = *(arr + min_index);

        *(arr + min_index) = *(arr + i);

        *(arr + i) = temp;

        }

}

bool is_prime(int num) {

        if(num == 1) {

                return false;

        }

        if(num == 2) {

                return true;
```

```
        }
        for(int fac = 2; fac < num; fac++) {
                if(num % fac == 0) {
                        return false;
                }
        }
}
                if(is_prime(*(arr + i))) {
                        cout << *(arr + i) << ": PRIME\n";
                }
                else {
                        cout << *(arr + i) << ": NOT PRIME\n";
                }
        }
        delete[] arr;
}
void sorting() {
        int *arr = new int[size];
        selection_sort(arr, size);

        cout << "sort in ascending or descending order? (a or d) :";
        cout << "after sorting: ";
        for(int i = 0; i < size; i++) {
                cout << "your linked list: ";
                cout << *(arr + i) << " ";
        }
        cout <<  endl;
```

```
        for(int i = 0; i < size; i++) {
                if(is_prime(*(arr + i))) {
                        cout << *(arr + i) << ":PRIME\n";
                }
                else {
                        cout << *(arr + i) << ":NOT PRIME\n";
                }
        }
        delete[] arr;
}
void ask_user() {
        int x;
        int y;
        x = 0;
        y = 0;
        string num;
        cout << "please enter a number: ";
        getline(cin, num);
        while(!(x)) {
                if(check_num(input)) {
                        cout << "invalid, enter a number: ";
                        getline(cin, num);
                }
                else {
                        x = 1;
                }
        }
}
```

```
void another_number(int &run) {
	cout << “do you want to enter another number? (y or n) :”;
	char input;
	cin >> input;

	if(input == y) {
		ask_user();
	}
	if(input == n) {
		sorting();
	}
}
int program() {
	int run = 1;
	while(run) {
		another_number(run);
	}
}
```

Testing:

Actual	Expected
please enter a number: x = jfkewas	print out an error message, reprompt the user for a new number
please enter a number: x = 5108	ask the user if they would like to enter another number
do you want another number? (y or n) input = ksmdne	print out an error message, reprompt the user asking to enter either y or n
do you want another number? (y or n) input = n	ask the user whether they would like the list sorted in ascending or descending order
do you want another number? (y or n) input = y	ask the user to enter a number to add to the list
sort in ascending or descending order? (a or d) input = kefni	print out an error message, reprompt the user asking to enter either a or d
sort in ascending or descending order? (a or d) input = a	print out the linked list in ascending order, print out the number of prime numbers in the list & prompt the user whether they would like to run the program again
sort in ascending or descending order? (a or d) input = d	print out the linked list in descending order, print out the number of prime numbers in the list & prompt the user whether they would like to run the program again
do you want to run the program again? input = jsfsdve	print out an error message, reprompt the user for an input either n or y
do you want to run the program again? input = n	quit the program
do you want to run the program again? input = y	ask the user to enter a number