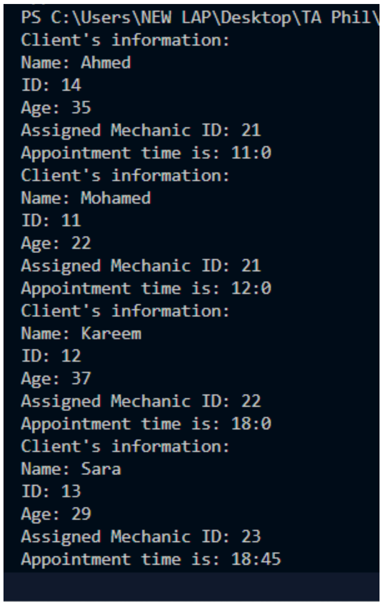
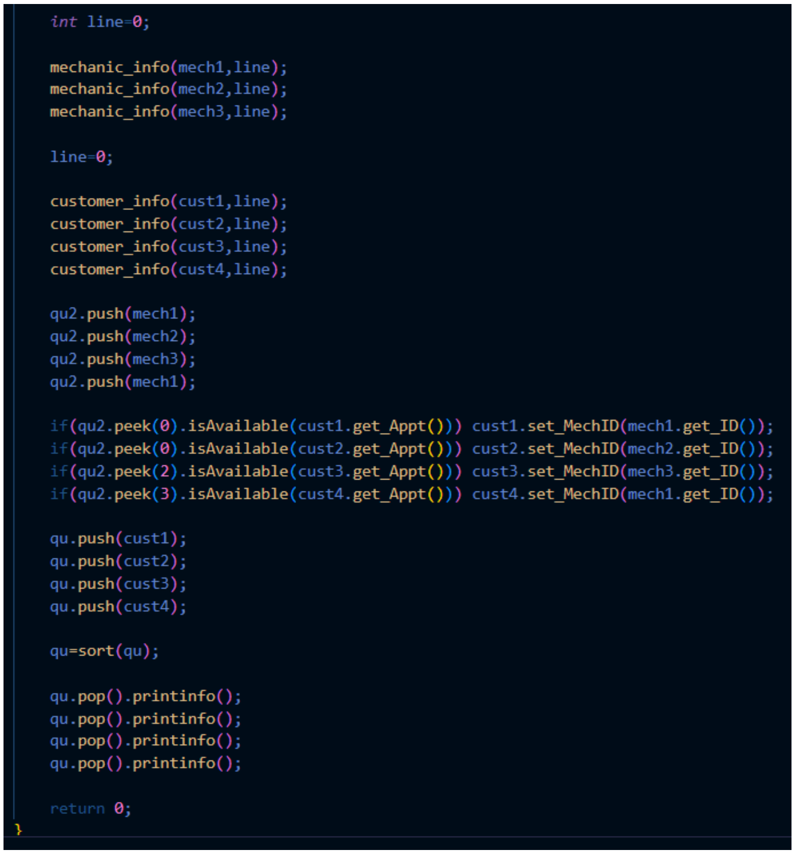
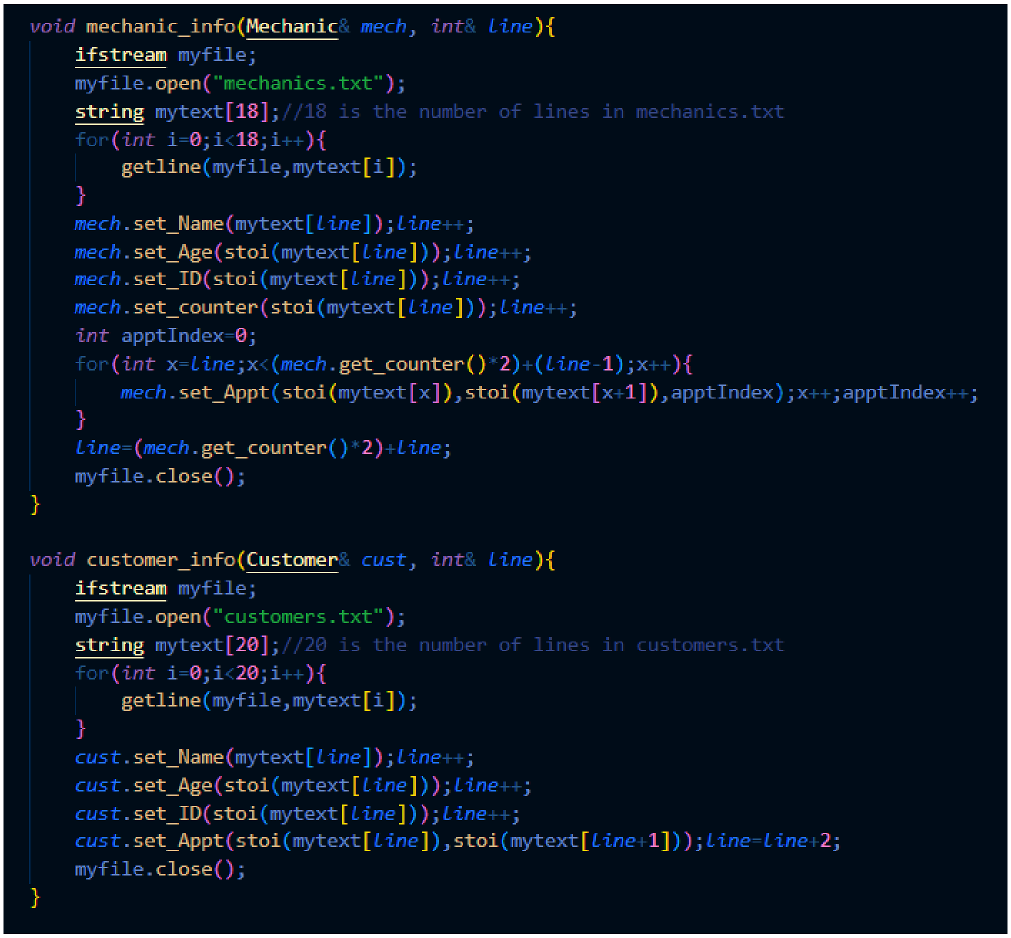


```
PS C:\Users\NEW LAP\Desktop\TA Phil\
Client's information:
Name: Ahmed
ID: 14
Age: 35
Assigned Mechanic ID: 21
Appointment time is: 11:0
Client's information:
Name: Mohamed
ID: 11
Age: 22
Assigned Mechanic ID: 21
Appointment time is: 12:0
Client's information:
Name: Kareem
ID: 12
Age: 37
Assigned Mechanic ID: 22
Appointment time is: 18:0
Client's information:
Name: Sara
ID: 13
Age: 29
Assigned Mechanic ID: 23
Appointment time is: 18:45
```



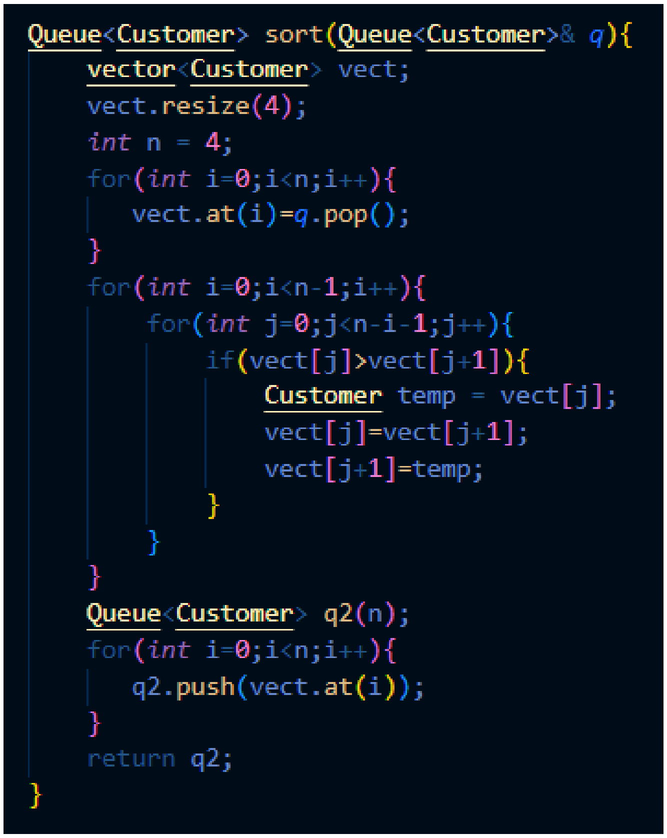
```
    int line=0;

    mechanic_info(mech1,line);
    mechanic_info(mech2,line);
    mechanic_info(mech3,line);

    line=0;

    customer_info(cust1,line);
    customer_info(cust2,line);
    customer_info(cust3,line);
    customer_info(cust4,line);

    qu2.push(mech1);
    qu2.push(mech2);
    qu2.push(mech3);
    qu2.push(mech1);

    if(qu2.peek(0).isAvailable(cust1.get_Appt())) cust1.set_MechID(mech1.get_ID());
    if(qu2.peek(0).isAvailable(cust2.get_Appt())) cust2.set_MechID(mech2.get_ID());
    if(qu2.peek(2).isAvailable(cust3.get_Appt())) cust3.set_MechID(mech3.get_ID());
    if(qu2.peek(3).isAvailable(cust4.get_Appt())) cust4.set_MechID(mech1.get_ID());

    qu.push(cust1);
    qu.push(cust2);
    qu.push(cust3);
    qu.push(cust4);

    qu=sort(qu);

    qu.pop().printinfo();
    qu.pop().printinfo();
    qu.pop().printinfo();
    qu.pop().printinfo();

    return 0;
}
```

This is the output of this code. This is the sorted queue of customers after being assigned to mechanics. The list was as follows: Mohamed, Kareem, Sara, and Ahmed. Mohamed was assigned to the first mechanic if they are available, and so on. So Mohamed was assigned to the first mechanic (ID 21), Kareem was assigned to the second mechanic (ID 22), Sara was assigned to the third mechanic (ID 23), and Ahmed was assigned to the first mechanic again (ID 21). Then the appointments were sorted using the reference to the overloaded operators in the customers class and this is the output.



```
void mechanic_info(Mechanic& mech, int& line){
    ifstream myfile;
    myfile.open("mechanics.txt");
    string mytext[18];//18 is the number of lines in mechanics.txt
    for(int i=0;i<18;i++){
        getline(myfile,mytext[i]);
    }
    mech.set_Name(mytext[line]);line++;
    mech.set_Age(stoi(mytext[line]));line++;
    mech.set_ID(stoi(mytext[line]));line++;
    mech.set_counter(stoi(mytext[line]));line++;
    int apptIndex=0;
    for(int x=line;x<(mech.get_counter()*2)+(line-1);x++){
        mech.set_Appt(stoi(mytext[x]),stoi(mytext[x+1]),apptIndex);x++;apptIndex++;
    }
    line=(mech.get_counter()*2)+line;
    myfile.close();
}

void customer_info(Customer& cust, int& line){
    ifstream myfile;
    myfile.open("customers.txt");
    string mytext[20];//20 is the number of lines in customers.txt
    for(int i=0;i<20;i++){
        getline(myfile,mytext[i]);
    }
    cust.set_Name(mytext[line]);line++;
    cust.set_Age(stoi(mytext[line]));line++;
    cust.set_ID(stoi(mytext[line]));line++;
    cust.set_Appt(stoi(mytext[line]),stoi(mytext[line+1]));line=line+2;
    myfile.close();
}
```

This is the bonus part, where these functions are called in main as seen in the second screenshot in this report. The algorithm relies on the number of lines. For the first customer or mechanic, the line number is 0. It increments and is passed by reference so that calling the function for the following customer would be start at the line right at the name of the customer that follows. In the case of mechanics, to read appointments goes through a loop to read the appointment and the line depends on multiplying the counter (i.e. the number of appointments) by two because every appointment has two lines assigned to it because an appointment is both minutes and hours. The file is closed at the end.



```
Queue<Customer> sort(Queue<Customer>& q){
    vector<Customer> vect;
    vect.resize(4);
    int n = 4;
    for(int i=0;i<n;i++){
        vect.at(i)=q.pop();
    }
    for(int i=0;i<n-1;i++){
        for(int j=0;j<n-i-1;j++){
            if(vect[j]>vect[j+1]){
                Customer temp = vect[j];
                vect[j]=vect[j+1];
                vect[j+1]=temp;
            }
        }
    }
    Queue<Customer> q2(n);
    for(int i=0;i<n;i++){
        q2.push(vect.at(i));
    }
    return q2;
}
```

This is the sorting algorithm I used which is bubble sorting algorithm. The queue is emptied in a vector, then the vector goes through the sorting code. Afterwards, the elements of the vector are pushed into a queue which is returned by the function.