

Activity No. <5.1>

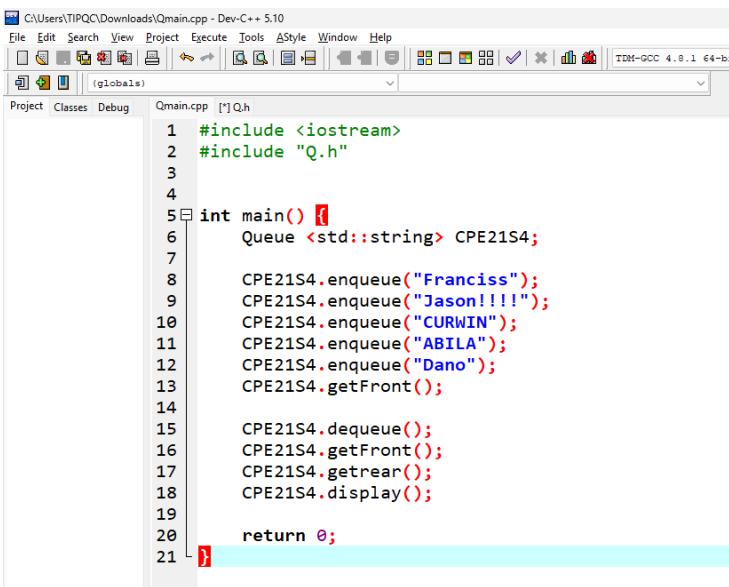
<Queue - Linked List Application>

Course Code: CPE010	Program: Computer Engineering
Course Title: Data Structures and Algorithms	Date Performed: 9/9/25
Section: CPE21S4	Date Submitted: 9/9/25
Name(s): Alcantara, Jason P.	Instructor: Engr. Jimlord Quejado

6. Output:

Syntax:

Main.cpp:

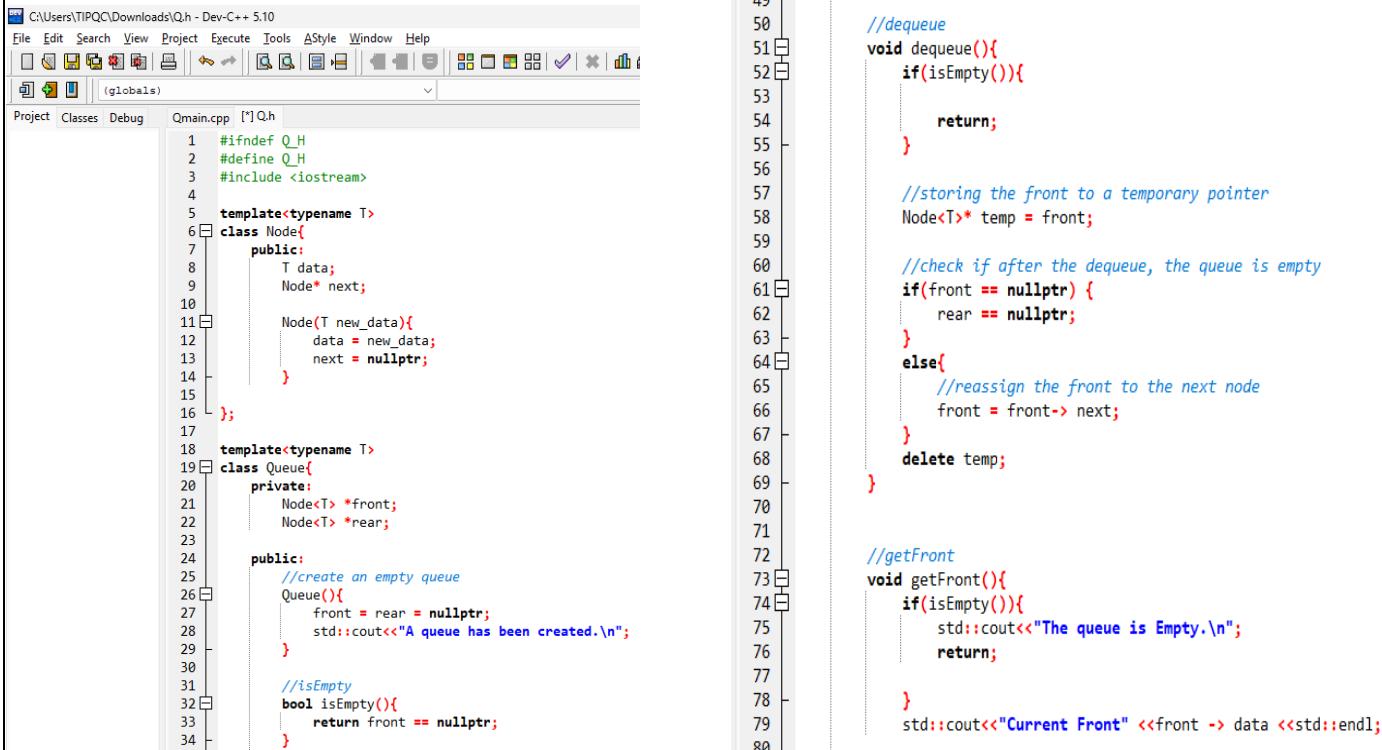


```

C:\Users\TIPQC\Downloads\Qmain.cpp - Dev-C++ 5.10
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug Qmain.cpp [*] Q.h
1 #include <iostream>
2 #include "Q.h"
3
4
5 int main() {
6     Queue<std::string> CPE21S4;
7
8     CPE21S4.enqueue("Franciss");
9     CPE21S4.enqueue("Jason!!!!");
10    CPE21S4.enqueue("CURWIN");
11    CPE21S4.enqueue("ABILA");
12    CPE21S4.enqueue("Dano");
13    CPE21S4.getFront();
14
15    CPE21S4.dequeue();
16    CPE21S4.getFront();
17    CPE21S4.getrear();
18    CPE21S4.display();
19
20    return 0;
21 }

```

Q.h file:



```

C:\Users\TIPQC\Downloads\Q.h - Dev-C++ 5.10
File Edit Search View Project Execute Tools AStyle Window Help
Project Classes Debug Qmain.cpp [*] Q.h
1 #ifndef Q_H
2 #define Q_H
3 #include <iostream>
4
5 template<typename T>
6 class Node{
7 public:
8     T data;
9     Node* next;
10
11     Node(T new_data){
12         data = new_data;
13         next = nullptr;
14     }
15 };
16
17 template<typename T>
18 class Queue{
19 private:
20     Node<T> *front;
21     Node<T> *rear;
22
23 public:
24     //create an empty queue
25     Queue(){
26         front = rear = nullptr;
27         std::cout<<"A queue has been created.\n";
28     }
29
30     //isEmpty
31     bool isEmpty(){
32         return front == nullptr;
33     }
34
35     //enqueue
36     void enqueue(T data){
37         Node<T> *newNode = new Node(data);
38         if(isEmpty()){
39             front = newNode;
40             rear = newNode;
41         } else{
42             rear->next = newNode;
43             rear = newNode;
44         }
45     }
46
47     //dequeue
48     void dequeue(){
49         if(isEmpty()){
50             return;
51         }
52
53         //storing the front to a temporary pointer
54         Node<T>* temp = front;
55
56         //check if after the dequeue, the queue is empty
57         if(front == nullptr){
58             rear == nullptr;
59         }
56
57         else{
58             //reassign the front to the next node
59             front = front->next;
60         }
61
62         delete temp;
63
64
65     }
66
67     //getFront
68     void getFront(){
69         if(isEmpty()){
70             std::cout<<"The queue is Empty.\n";
71         }
72
73     }
74
75     //display
76     void display(){
77         Node<T> *temp = front;
78
79         while(temp != nullptr){
80             std::cout<<temp->data<<std::endl;
81             temp = temp->next;
82         }
83     }
84
85     //getrear
86     void getrear(){
87         if(isEmpty()){
88             std::cout<<"The queue is Empty.\n";
89         }
90         else{
91             std::cout<<rear->data<<std::endl;
92         }
93     }
94
95     //size
96     int size(){
97         Node<T> *temp = front;
98         int count = 0;
99
100        while(temp != nullptr){
101            count++;
102            temp = temp->next;
103        }
104
105        return count;
106    }
107
108    //front
109    T frontData(){
110        if(isEmpty()){
111            std::cout<<"The queue is Empty.\n";
112            return -1;
113        }
114        else{
115            return front->data;
116        }
117    }
118
119    //rear
120    T rearData(){
121        if(isEmpty()){
122            std::cout<<"The queue is Empty.\n";
123            return -1;
124        }
125        else{
126            return rear->data;
127        }
128    }
129
130    //display
131    void display(){
132        Node<T> *temp = front;
133
134        while(temp != nullptr){
135            std::cout<<temp->data<<std::endl;
136            temp = temp->next;
137        }
138    }
139
140    //getFront
141    T getFront(){
142        if(isEmpty()){
143            std::cout<<"The queue is Empty.\n";
144            return -1;
145        }
146        else{
147            return front->data;
148        }
149    }
150
151    //getrear
152    T getrear(){
153        if(isEmpty()){
154            std::cout<<"The queue is Empty.\n";
155            return -1;
156        }
157        else{
158            return rear->data;
159        }
160    }
161
162    //size
163    int size(){
164        Node<T> *temp = front;
165        int count = 0;
166
167        while(temp != nullptr){
168            count++;
169            temp = temp->next;
170        }
171
172        return count;
173    }
174
175    //front
176    T frontData(){
177        if(isEmpty()){
178            std::cout<<"The queue is Empty.\n";
179            return -1;
180        }
181        else{
182            return front->data;
183        }
184    }
185
186    //rear
187    T rearData(){
188        if(isEmpty()){
189            std::cout<<"The queue is Empty.\n";
190            return -1;
191        }
192        else{
193            return rear->data;
194        }
195    }
196
197    //display
198    void display(){
199        Node<T> *temp = front;
200
201        while(temp != nullptr){
202            std::cout<<temp->data<<std::endl;
203            temp = temp->next;
204        }
205    }
206
207    //getFront
208    T getFront(){
209        if(isEmpty()){
210            std::cout<<"The queue is Empty.\n";
211            return -1;
212        }
213        else{
214            return front->data;
215        }
216    }
217
218    //getrear
219    T getrear(){
220        if(isEmpty()){
221            std::cout<<"The queue is Empty.\n";
222            return -1;
223        }
224        else{
225            return rear->data;
226        }
227    }
228
229    //size
230    int size(){
231        Node<T> *temp = front;
232        int count = 0;
233
234        while(temp != nullptr){
235            count++;
236            temp = temp->next;
237        }
238
239        return count;
240    }
241
242    //front
243    T frontData(){
244        if(isEmpty()){
245            std::cout<<"The queue is Empty.\n";
246            return -1;
247        }
248        else{
249            return front->data;
250        }
251    }
252
253    //rear
254    T rearData(){
255        if(isEmpty()){
256            std::cout<<"The queue is Empty.\n";
257            return -1;
258        }
259        else{
260            return rear->data;
261        }
262    }
263
264    //display
265    void display(){
266        Node<T> *temp = front;
267
268        while(temp != nullptr){
269            std::cout<<temp->data<<std::endl;
270            temp = temp->next;
271        }
272    }
273
274    //getFront
275    T getFront(){
276        if(isEmpty()){
277            std::cout<<"The queue is Empty.\n";
278            return -1;
279        }
280        else{
281            return front->data;
282        }
283    }
284
285    //getrear
286    T getrear(){
287        if(isEmpty()){
288            std::cout<<"The queue is Empty.\n";
289            return -1;
290        }
291        else{
292            return rear->data;
293        }
294    }
295
296    //size
297    int size(){
298        Node<T> *temp = front;
299        int count = 0;
300
301        while(temp != nullptr){
302            count++;
303            temp = temp->next;
304        }
305
306        return count;
307    }
308
309    //front
310    T frontData(){
311        if(isEmpty()){
312            std::cout<<"The queue is Empty.\n";
313            return -1;
314        }
315        else{
316            return front->data;
317        }
318    }
319
320    //rear
321    T rearData(){
322        if(isEmpty()){
323            std::cout<<"The queue is Empty.\n";
324            return -1;
325        }
326        else{
327            return rear->data;
328        }
329    }
330
331    //display
332    void display(){
333        Node<T> *temp = front;
334
335        while(temp != nullptr){
336            std::cout<<temp->data<<std::endl;
337            temp = temp->next;
338        }
339    }
340
341    //getFront
342    T getFront(){
343        if(isEmpty()){
344            std::cout<<"The queue is Empty.\n";
345            return -1;
346        }
347        else{
348            return front->data;
349        }
350    }
351
352    //getrear
353    T getrear(){
354        if(isEmpty()){
355            std::cout<<"The queue is Empty.\n";
356            return -1;
357        }
358        else{
359            return rear->data;
360        }
361    }
362
363    //size
364    int size(){
365        Node<T> *temp = front;
366        int count = 0;
367
368        while(temp != nullptr){
369            count++;
370            temp = temp->next;
371        }
372
373        return count;
374    }
375
376    //front
377    T frontData(){
378        if(isEmpty()){
379            std::cout<<"The queue is Empty.\n";
380            return -1;
381        }
382        else{
383            return front->data;
384        }
385    }
386
387    //rear
388    T rearData(){
389        if(isEmpty()){
390            std::cout<<"The queue is Empty.\n";
391            return -1;
392        }
393        else{
394            return rear->data;
395        }
396    }
397
398    //display
399    void display(){
400        Node<T> *temp = front;
401
402        while(temp != nullptr){
403            std::cout<<temp->data<<std::endl;
404            temp = temp->next;
405        }
406    }
407
408    //getFront
409    T getFront(){
410        if(isEmpty()){
411            std::cout<<"The queue is Empty.\n";
412            return -1;
413        }
414        else{
415            return front->data;
416        }
417    }
418
419    //getrear
420    T getrear(){
421        if(isEmpty()){
422            std::cout<<"The queue is Empty.\n";
423            return -1;
424        }
425        else{
426            return rear->data;
427        }
428    }
429
430    //size
431    int size(){
432        Node<T> *temp = front;
433        int count = 0;
434
435        while(temp != nullptr){
436            count++;
437            temp = temp->next;
438        }
439
440        return count;
441    }
442
443    //front
444    T frontData(){
445        if(isEmpty()){
446            std::cout<<"The queue is Empty.\n";
447            return -1;
448        }
449        else{
450            return front->data;
451        }
452    }
453
454    //rear
455    T rearData(){
456        if(isEmpty()){
457            std::cout<<"The queue is Empty.\n";
458            return -1;
459        }
460        else{
461            return rear->data;
462        }
463    }
464
465    //display
466    void display(){
467        Node<T> *temp = front;
468
469        while(temp != nullptr){
470            std::cout<<temp->data<<std::endl;
471            temp = temp->next;
472        }
473    }
474
475    //getFront
476    T getFront(){
477        if(isEmpty()){
478            std::cout<<"The queue is Empty.\n";
479            return -1;
480        }
481        else{
482            return front->data;
483        }
484    }
485
486    //getrear
487    T getrear(){
488        if(isEmpty()){
489            std::cout<<"The queue is Empty.\n";
490            return -1;
491        }
492        else{
493            return rear->data;
494        }
495    }
496
497    //size
498    int size(){
499        Node<T> *temp = front;
500        int count = 0;
501
502        while(temp != nullptr){
503            count++;
504            temp = temp->next;
505        }
506
507        return count;
508    }
509
510    //front
511    T frontData(){
512        if(isEmpty()){
513            std::cout<<"The queue is Empty.\n";
514            return -1;
515        }
516        else{
517            return front->data;
518        }
519    }
520
521    //rear
522    T rearData(){
523        if(isEmpty()){
524            std::cout<<"The queue is Empty.\n";
525            return -1;
526        }
527        else{
528            return rear->data;
529        }
530    }
531
532    //display
533    void display(){
534        Node<T> *temp = front;
535
536        while(temp != nullptr){
537            std::cout<<temp->data<<std::endl;
538            temp = temp->next;
539        }
540    }
541
542    //getFront
543    T getFront(){
544        if(isEmpty()){
545            std::cout<<"The queue is Empty.\n";
546            return -1;
547        }
548        else{
549            return front->data;
550        }
551    }
552
553    //getrear
554    T getrear(){
555        if(isEmpty()){
556            std::cout<<"The queue is Empty.\n";
557            return -1;
558        }
559        else{
560            return rear->data;
561        }
562    }
563
564    //size
565    int size(){
566        Node<T> *temp = front;
567        int count = 0;
568
569        while(temp != nullptr){
570            count++;
571            temp = temp->next;
572        }
573
574        return count;
575    }
576
577    //front
578    T frontData(){
579        if(isEmpty()){
580            std::cout<<"The queue is Empty.\n";
581            return -1;
582        }
583        else{
584            return front->data;
585        }
586    }
587
588    //rear
589    T rearData(){
590        if(isEmpty()){
591            std::cout<<"The queue is Empty.\n";
592            return -1;
593        }
594        else{
595            return rear->data;
596        }
597    }
598
599    //display
600    void display(){
601        Node<T> *temp = front;
602
603        while(temp != nullptr){
604            std::cout<<temp->data<<std::endl;
605            temp = temp->next;
606        }
607    }
608
609    //getFront
610    T getFront(){
611        if(isEmpty()){
612            std::cout<<"The queue is Empty.\n";
613            return -1;
614        }
615        else{
616            return front->data;
617        }
618    }
619
620    //getrear
621    T getrear(){
622        if(isEmpty()){
623            std::cout<<"The queue is Empty.\n";
624            return -1;
625        }
626        else{
627            return rear->data;
628        }
629    }
630
631    //size
632    int size(){
633        Node<T> *temp = front;
634        int count = 0;
635
636        while(temp != nullptr){
637            count++;
638            temp = temp->next;
639        }
640
641        return count;
642    }
643
644    //front
645    T frontData(){
646        if(isEmpty()){
647            std::cout<<"The queue is Empty.\n";
648            return -1;
649        }
650        else{
651            return front->data;
652        }
653    }
654
655    //rear
656    T rearData(){
657        if(isEmpty()){
658            std::cout<<"The queue is Empty.\n";
659            return -1;
660        }
661        else{
662            return rear->data;
663        }
664    }
665
666    //display
667    void display(){
668        Node<T> *temp = front;
669
670        while(temp != nullptr){
671            std::cout<<temp->data<<std::endl;
672            temp = temp->next;
673        }
674    }
675
676    //getFront
677    T getFront(){
678        if(isEmpty()){
679            std::cout<<"The queue is Empty.\n";
680            return -1;
681        }
682        else{
683            return front->data;
684        }
685    }
686
687    //getrear
688    T getrear(){
689        if(isEmpty()){
690            std::cout<<"The queue is Empty.\n";
691            return -1;
692        }
693        else{
694            return rear->data;
695        }
696    }
697
698    //size
699    int size(){
700        Node<T> *temp = front;
701        int count = 0;
702
703        while(temp != nullptr){
704            count++;
705            temp = temp->next;
706        }
707
708        return count;
709    }
710
711    //front
712    T frontData(){
713        if(isEmpty()){
714            std::cout<<"The queue is Empty.\n";
715            return -1;
716        }
717        else{
718            return front->data;
719        }
720    }
721
722    //rear
723    T rearData(){
724        if(isEmpty()){
725            std::cout<<"The queue is Empty.\n";
726            return -1;
727        }
728        else{
729            return rear->data;
730        }
731    }
732
733    //display
734    void display(){
735        Node<T> *temp = front;
736
737        while(temp != nullptr){
738            std::cout<<temp->data<<std::endl;
739            temp = temp->next;
740        }
741    }
742
743    //getFront
744    T getFront(){
745        if(isEmpty()){
746            std::cout<<"The queue is Empty.\n";
747            return -1;
748        }
749        else{
750            return front->data;
751        }
752    }
753
754    //getrear
755    T getrear(){
756        if(isEmpty()){
757            std::cout<<"The queue is Empty.\n";
758            return -1;
759        }
760        else{
761            return rear->data;
762        }
763    }
764
765    //size
766    int size(){
767        Node<T> *temp = front;
768        int count = 0;
769
770        while(temp != nullptr){
771            count++;
772            temp = temp->next;
773        }
774
775        return count;
776    }
777
778    //front
779    T frontData(){
780        if(isEmpty()){
781            std::cout<<"The queue is Empty.\n";
782            return -1;
783        }
784        else{
785            return front->data;
786       }
787    }
788
789    //rear
790    T rearData(){
791        if(isEmpty()){
792            std::cout<<"The queue is Empty.\n";
793            return -1;
794       }
795        else{
796            return rear->data;
797       }
798    }
799
800    //display
801    void display(){
802        Node<T> *temp = front;
803
804        while(temp != nullptr){
805            std::cout<<temp->data<<std::endl;
806            temp = temp->next;
807        }
808    }
809
810    //getFront
811    T getFront(){
812        if(isEmpty()){
813            std::cout<<"The queue is Empty.\n";
814            return -1;
815       }
816        else{
817            return front->data;
818       }
819    }
820
821    //getrear
822    T getrear(){
823        if(isEmpty()){
824            std::cout<<"The queue is Empty.\n";
825            return -1;
826       }
827        else{
828            return rear->data;
829       }
830    }
831
832    //size
833    int size(){
834        Node<T> *temp = front;
835        int count = 0;
836
837        while(temp != nullptr){
838            count++;
839            temp = temp->next;
840       }
841
842        return count;
843    }
844
845    //front
846    T frontData(){
847        if(isEmpty()){
848            std::cout<<"The queue is Empty.\n";
849            return -1;
850       }
851        else{
852            return front->data;
853       }
854    }
855
856    //rear
857    T rearData(){
858        if(isEmpty()){
859            std::cout<<"The queue is Empty.\n";
860            return -1;
861       }
862        else{
863            return rear->data;
864       }
865    }
866
867    //display
868    void display(){
869        Node<T> *temp = front;
870
871        while(temp != nullptr){
872            std::cout<<temp->data<<std::endl;
873            temp = temp->next;
874       }
875    }
876
877    //getFront
878    T getFront(){
879        if(isEmpty()){
880            std::cout<<"The queue is Empty.\n";
881            return -1;
882       }
883        else{
884            return front->data;
885       }
886    }
887
888    //getrear
889    T getrear(){
890        if(isEmpty()){
891            std::cout<<"The queue is Empty.\n";
892            return -1;
893       }
894        else{
895            return rear->data;
896       }
897    }
898
899    //size
900    int size(){
901        Node<T> *temp = front;
902        int count = 0;
903
904        while(temp != nullptr){
905            count++;
906            temp = temp->next;
907       }
908
909        return count;
910    }
911
912    //front
913    T frontData(){
914        if(isEmpty()){
915            std::cout<<"The queue is Empty.\n";
916            return -1;
917       }
918        else{
919            return front->data;
920       }
921    }
922
923    //rear
924    T rearData(){
925        if(isEmpty()){
926            std::cout<<"The queue is Empty.\n";
927            return -1;
928       }
929        else{
930            return rear->data;
931       }
932    }
933
934    //display
935    void display(){
936        Node<T> *temp = front;
937
938        while(temp != nullptr){
939            std::cout<<temp->data<<std::endl;
940            temp = temp->next;
941       }
942    }
943
944    //getFront
945    T getFront(){
946        if(isEmpty()){
947            std::cout<<"The queue is Empty.\n";
948            return -1;
949       }
950        else{
951            return front->data;
952       }
953    }
954
955    //getrear
956    T getrear(){
957        if(isEmpty()){
958            std::cout<<"The queue is Empty.\n";
959            return -1;
960       }
961        else{
962            return rear->data;
963       }
964    }
965
966    //size
967    int size(){
968        Node<T> *temp = front;
969        int count = 0;
970
971        while(temp != nullptr){
972            count++;
973            temp = temp->next;
974       }
975
976        return count;
977    }
978
979    //front
980    T frontData(){
981        if(isEmpty()){
982            std::cout<<"The queue is Empty.\n";
983            return -1;
984       }
985        else{
986            return front->data;
987       }
988    }
989
990    //rear
991    T rearData(){
992        if(isEmpty()){
993            std::cout<<"The queue is Empty.\n";
994            return -1;
995       }
996        else{
997            return rear->data;
998       }
999    }
1000
1001    //display
1002    void display(){
1003        Node<T> *temp = front;
1004
1005        while(temp != nullptr){
1006            std::cout<<temp->data<<std::endl;
1007            temp = temp->next;
1008       }
1009    }
1010
1011    //getFront
1012    T getFront(){
1013        if(isEmpty()){
1014            std::cout<<"The queue is Empty.\n";
1015            return -1;
1016       }
1017        else{
1018            return front->data;
1019       }
1020    }
1021
1022    //getrear
1023    T getrear(){
1024        if(isEmpty()){
1025            std::cout<<"The queue is Empty.\n";
1026            return -1;
1027       }
1028        else{
1029            return rear->data;
1030       }
1031    }
1032
1033    //size
1034    int size(){
1035        Node<T> *temp = front;
1036        int count = 0;
1037
1038        while(temp != nullptr){
1039            count++;
1040            temp = temp->next;
1041       }
1042
1043        return count;
1044    }
1045
1046    //front
1047    T frontData(){
1048        if(isEmpty()){
1049            std::cout<<"The queue is Empty.\n";
1050            return -1;
1051       }
1052        else{
1053            return front->data;
1054       }
1055    }
1056
1057    //rear
1058    T rearData(){
1059        if(isEmpty()){
1060            std::cout<<"The queue is Empty.\n";
1061            return -1;
1062       }
1063        else{
1064            return rear->data;
1065       }
1066    }
1067
1068    //display
1069    void display(){
1070        Node<T> *temp = front;
1071
1072        while(temp != nullptr){
1073            std::cout<<temp->data<<std::endl;
1074            temp = temp->next;
1075       }
1076    }
1077
1078    //getFront
1079    T getFront(){
1080        if(isEmpty()){
1081            std::cout<<"The queue is Empty.\n";
1082            return -1;
1083       }
1084        else{
1085            return front->data;
1086       }
1087    }
1088
1089    //getrear
1090    T getrear(){
1091        if(isEmpty()){
1092            std::cout<<"The queue is Empty.\n";
1093            return -1;
1094       }
1095        else{
1096            return rear->data;
1097       }
1098    }
1099
1100    //size
1101    int size(){
1102        Node<T> *temp = front;
1103        int count = 0;
1104
1105        while(temp != nullptr){
1106            count++;
1107            temp = temp->next;
1108       }
1109
1110        return count;
1111    }
1112
1113    //front
1114    T frontData(){
1115        if(isEmpty()){
1116            std::cout<<"The queue is Empty.\n";
1117            return -1;
1118       }
1119        else{
1120            return front->data;
1121       }
1122    }
1123
1124    //rear
1125    T rearData(){
1126        if(isEmpty()){
1127            std::cout<<"The queue is Empty.\n";
1128            return -1;
1129       }
1130        else{
1131            return rear->data;
1132       }
1133    }
1134
1135    //display
1136    void display(){
1137        Node<T> *temp = front;
1138
1139        while(temp != nullptr){
1140            std::cout<<temp->data<<std::endl;
1141            temp = temp->next;
1142       }
1143    }
1144
1145    //getFront
1146    T getFront(){
1147        if(isEmpty()){
1148            std::cout<<"The queue is Empty.\n";
1149            return -1;
1150       }
1151        else{
1152            return front->data;
1153       }
1154    }
1155
1156    //getrear
1157    T getrear(){
1158        if(isEmpty()){
1159            std::cout<<"The queue is Empty.\n";
1160            return -1;
1161       }
1162        else{
1163            return rear->data;
1164       }
1165    }
1166
1167    //size
1168    int size(){
1169        Node<T> *temp = front;
1170        int count = 0;
1171
1172        while(temp != nullptr){
1173            count++;
1174            temp = temp->next;
1175       }
1176
1177        return count;
1178    }
1179
1180    //front
1181    T frontData(){
1182        if(isEmpty()){
1183            std::cout<<"The queue is Empty.\n";
1184            return -1;
1185       }
1186        else{
1187            return front->data;
1188       }
1189    }
1190
1191    //rear
1192    T rearData(){
1193        if(isEmpty()){
1194            std::cout<<"The queue is Empty.\n";
1195            return -1;
1196       }
1197        else{
1198            return rear->data;
1199       }
1200    }
1201
1202    //display
1203    void display(){
1204        Node<T> *temp = front;
1205
1206        while(temp != nullptr){
1207            std::cout<<temp->data<<std::endl;
1208            temp = temp->next;
1209       }
1210    }
1211
1212    //getFront
1213    T getFront(){
1214        if(isEmpty()){
1215            std::cout<<"The queue is Empty.\n";
1216            return -1;
1217       }
1218        else{
1219            return front->data;
1220       }
1221    }
1222
1223    //getrear
1224    T getrear(){
1225        if(isEmpty()){
1226            std::cout<<"The queue is Empty.\n";
1227            return -1;
1228       }
1229        else{
1230            return rear->data;
1231       }
1232    }
1233
1234    //size
1235    int size(){
1236        Node<T> *temp = front;
1237        int count = 0;
1238
1239        while(temp != nullptr){
1240            count++;
1241            temp = temp->next;
1242       }
1243
1244        return count;
1245    }
1246
1247    //front
1248    T frontData(){
1249        if(isEmpty()){
1250            std::cout<<"The queue is Empty.\n";
1251            return -1;
1252       }
1253        else{
1254            return front->data;
1255       }
1256    }
1257
1258    //rear
1259    T rearData(){
1260        if(isEmpty()){
1261            std::cout<<"The queue is Empty.\n";
1262            return -1;
1263       }
1264        else{
1265            return rear->data;
1266       }
1267    }
1268
1269    //display
1270    void display(){
1271        Node<T> *temp = front;
1272
1273        while(temp != nullptr){
1274            std::cout<<temp->data<<std::endl;
1275            temp = temp->next;
1276       }
1277    }
1278
1279    //getFront
1280    T getFront(){
1281        if(isEmpty()){
1282            std::cout<<"The queue is Empty.\n";
1283            return -1;
1284       }
12
```

```

80
81
82
83     }
84     //getrear
85     void getrear(){
86         if(isEmpty()){
87             std::cout << "The queue is empty.\n";
88             return;
89         }
90         std::cout << "Current Rear: " << rear->data << std::endl;
91     }
92     //display
93     void display(){
94         if(isEmpty()){
95             std::cout << "The queue is Empty.\n";
96             return;
97         }
98         Node<T> *temp=Front;
99         while(temp !=nullptr){
100             std::cout << temp->data << " ";
101             temp = temp->next;
102         }
103         std::cout << std::endl;
104     }
105
106     //to deallocate memory
107     ~Queue(){
108         while(!isEmpty()){
109             dequeue();
110         }
111     }
112
113 };
114
115
116
117 #endif

```

Output:

```

C:\Users\TIPQC\Downloads\Q X + ▾
A queue has been created.
Enqueue to an empty queue.
successfull enqueue,
successfull enqueue,
successfull enqueue,
successfull enqueue,
Current FrontFranciss
Current FrontJason!!!!
Current Rear: Dano
Jason!!!! CURWIN ABILA Dano
-----
Process exited after 0.01483 seconds with return value 0
Press any key to continue . . .

```

7. Supplementary Activity

8. Conclusion:

In this laboratory activity we learned the how to create a queue. We always need to check if the data is empty because if we don't check the data if its empty the program will faced an error. And I learned how to used an temporary pointer and how to reassign the front node to the rear node. In my general though about this im still get confused a little bit that's why I need to read, practice and understand more about this activity so next I will not get confused on how to created a queue but still in this day I learned a lot today.

9. Assessment Rubric