

Presentación:

**Practica 8. Cola dinámica**

**Nombre: Farfán de León José Osvaldo**

**Código: 214796622**

**Carrera: Ingeniería en computación**

**Materia: Seminario de Estructura de Datos l**

**Profesor: Julio Esteban Valdés López**

**Sección: “D13”**

**Fecha de entrega: 24/11/2020**

Introducción:

Objetivo de la practica: Implementar una cola dinámica

Análisis del problema: Crear un nodo el cual contenga su campo valor y su campo siguiente, inicializar su cabecera con frente y final en null

Datos de entrada y precondiciones: El usuario ingresara los datos de la cola.

Datos o elementos de salida: Cola.

Resultados obtenidos (captura de pantalla):

Código fuente en C++:

1 #include <iostream>

2 #include <stdlib.h>

3 **using namespace std**;

4

5 **struct** Nodo{

6 **int** dato;

7 Nodo \*siguiente;

8 };

9

10 **int** vacia(Nodo \*&frente){

11 **if**(frente == NULL){

12 **return** 1;

13 }**else**{

14 **return** 0;

15 }

16 }

17

18 **void** encolar(**int** num, Nodo \*&frente, Nodo \*&**final**){

19 **int** aux;

20 Nodo \*nuevo\_nodo= **new** Nodo();

21 nuevo\_nodo->dato = num;

22 nuevo\_nodo->siguiente = NULL;

23 **if**(**final** != NULL){

24 **final**->siguiente = nuevo\_nodo;

25 }

26 **final** = nuevo\_nodo;

27 **if**(frente == NULL){

28 frente = nuevo\_nodo;

29 }

30 }

31

32 **void** imprimir(Nodo \*&frente){

33 Nodo \*temp;

34 temp=frente;

35 **while**(temp != NULL){

36 **cout**<<temp->dato<<" ";

37 temp = temp->siguiente;

38 }

39 **cout**<<""<<**endl**;

40 };

41

42 **void** desencolar(Nodo \*&frente, Nodo \*&**final**){

43 Nodo \*aux=frente;

44 **if**(vacia(frente)){

45 **cout**<<"Insuficiencia de datos"<<**endl**;

46 system("pause");

47 **return**;

48 }

49 aux = frente;

50 frente = frente->siguiente;

51 **if**(frente == NULL){

52 **final** == NULL;

53 }

54 **delete** aux;

55 }

56

57 **int** main(){

58 **int** opc, num;

59 Nodo \*frente=NULL;

60 Nodo \***final**=NULL;

61 **do**{

62 system("cls");

63 imprimir(frente);

64 **cout**<<""<<**endl**;

65 **cout** << "Menu:" << **endl**;

66 **cout** << " 1): Vacia" << **endl**;

67 **cout** << " 2): Encolar" << **endl**;

68 **cout** << " 3): Desencolar" << **endl**;

69 **cout** << " 4): Frente" << **endl**;

70 **cout** << " 5): Imprimir" << **endl**;

71 **cout** << " 0): SALIR"<<**endl**;

72 **cout**<<"Ingrese una opcion valida: ";

73 **cin** >> opc;

74 **switch**(opc){

75 **case** 1:

76 **cout**<<vacia(frente)<<**endl**;system("pause");

77 **break**;

78 **case** 2:

79 **cout**<<"Ingrese el elemento a agregar: ";

80 **cin**>>num;

81 encolar(num, frente, **final**);

82 **break**;

83 **case** 3:

84 desencolar(frente, final);

85 break;

86 case 4:

87 if(frente != NULL){

88 cout<<frente->dato<<endl;system("pause");

89 }else{

90 cout<<"Insuficiencia de datos"<<endl;

91 system("pause");

92 }

93 break;

94 case 5:

95 imprimir(frente);system("pause");

96 break;

97 case 0:

98 cout<<"Fin del programa"<<endl;

99 break;

100 }

101 }while(opc!=0);

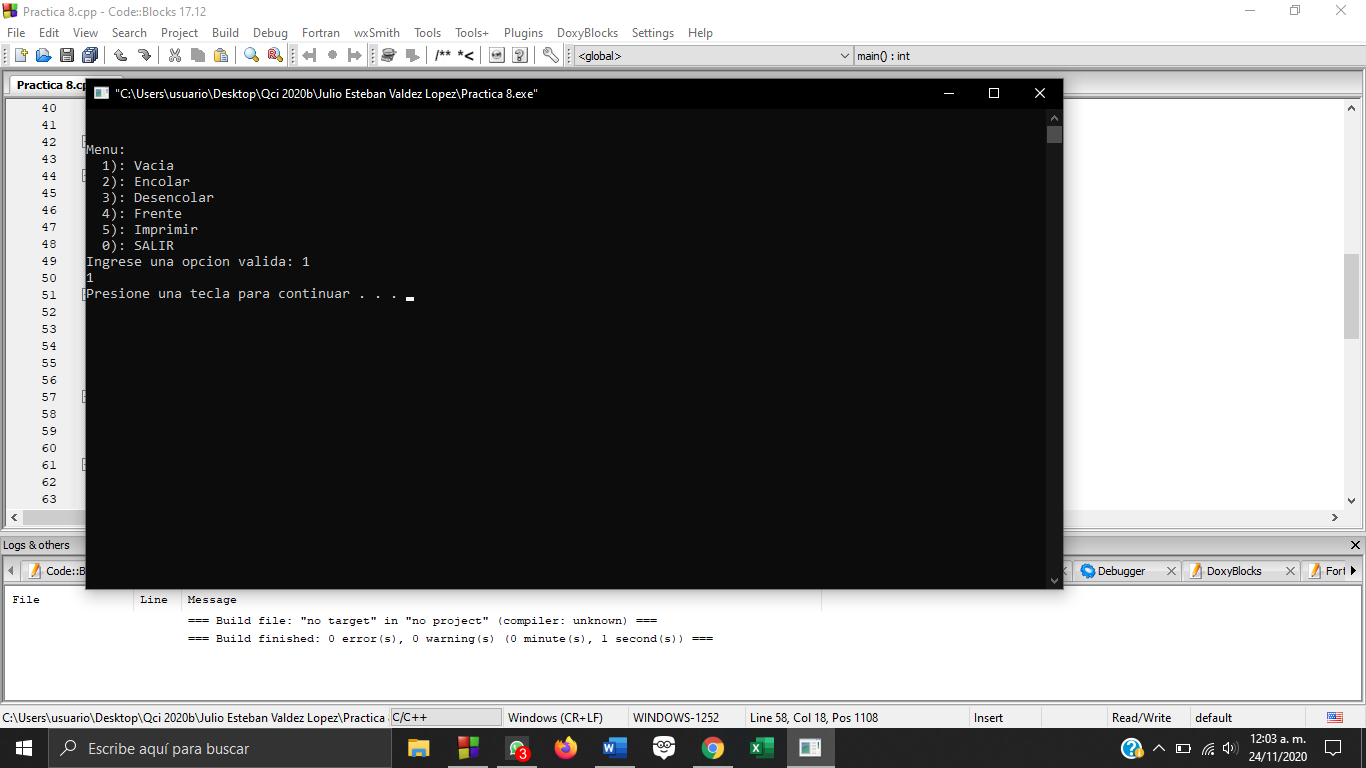
102

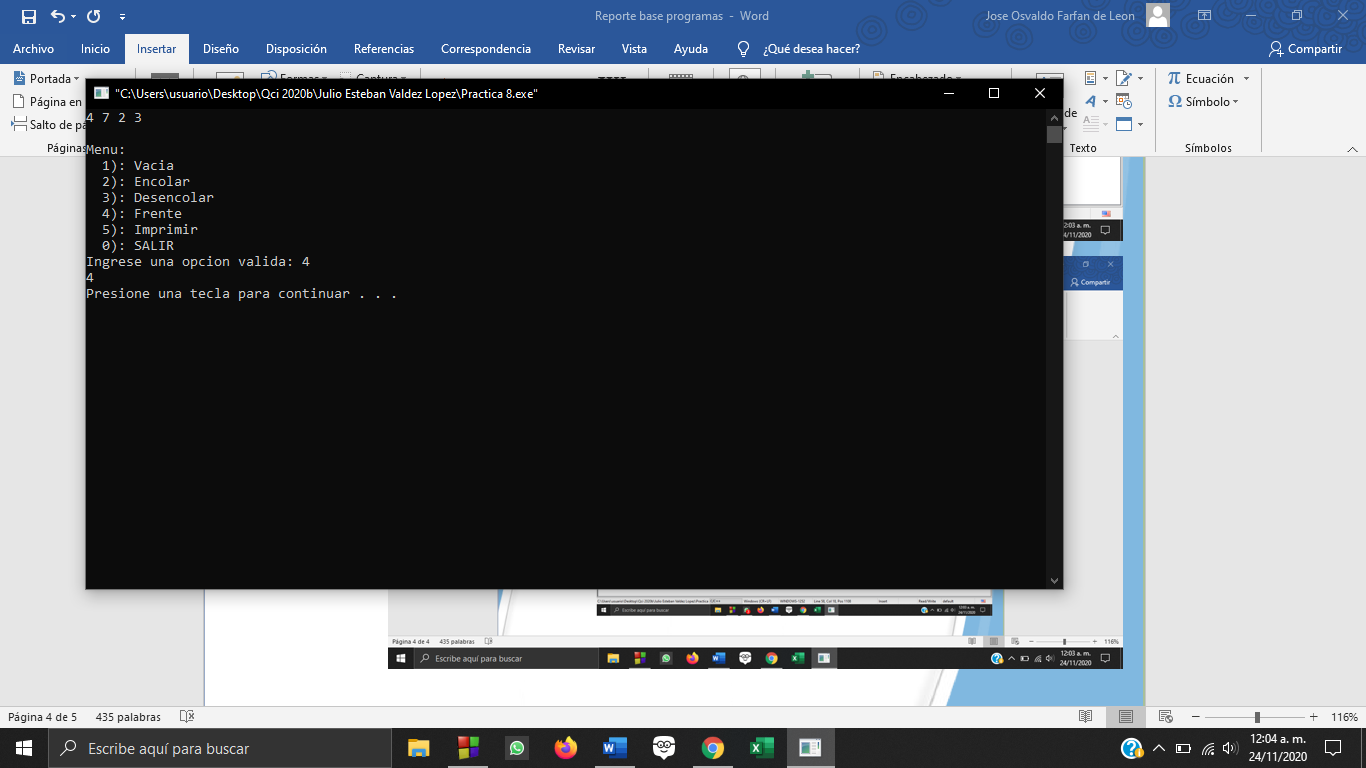
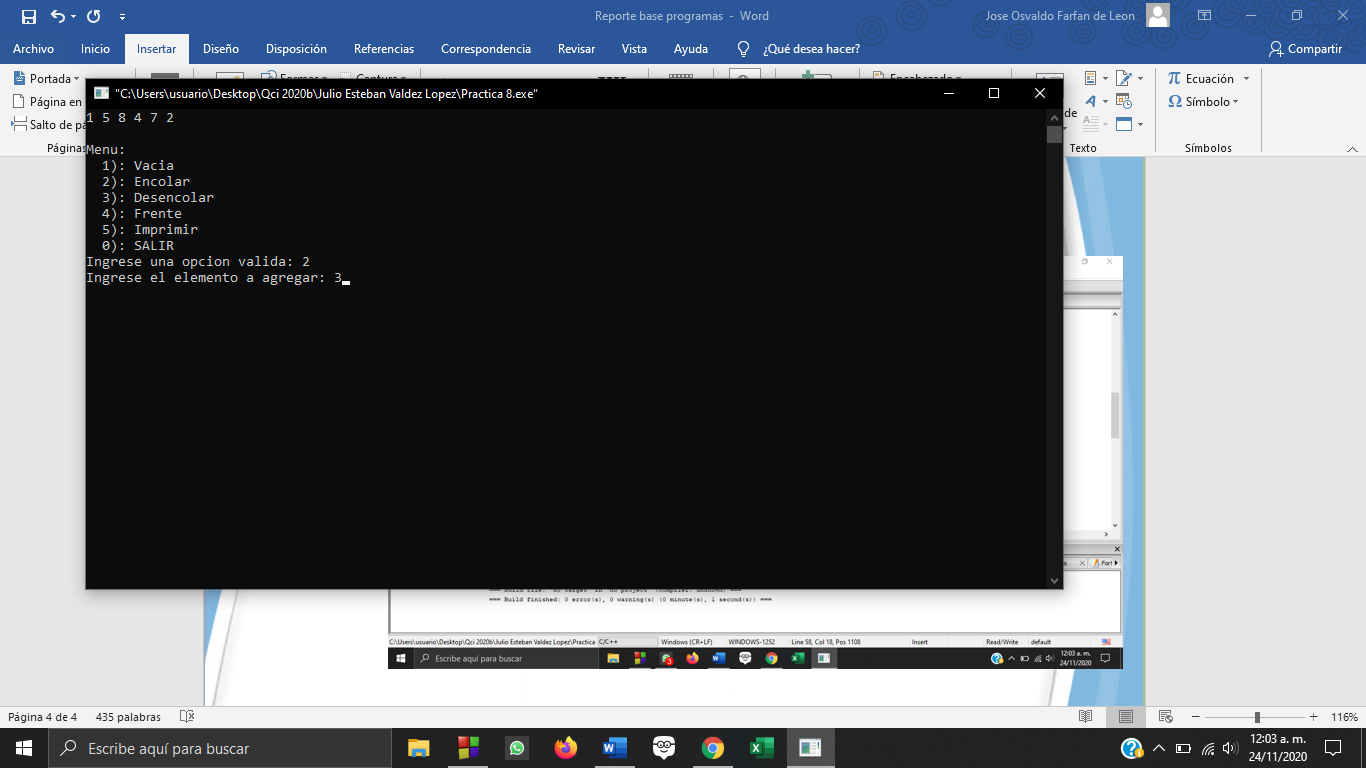
103 system("cls");

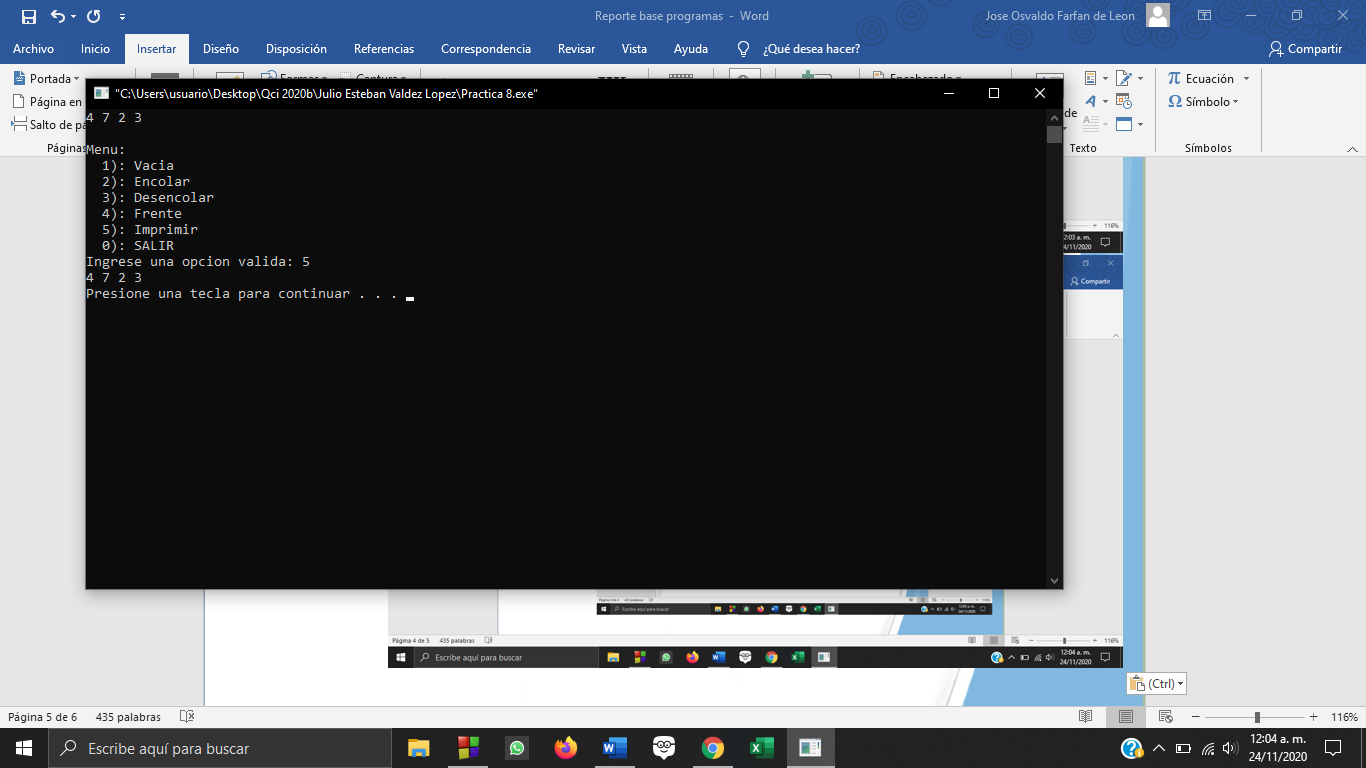
104 return 0;

105 }

Pantalla de ejecucion:







Conclucion:

Asegurarte de al principio del programa inicializar lo que seria el frente y el final en null, también al eliminar un nodo guardar la memoria del nodo a eliminar para poder liberar la memoria que esta esta ocupando.