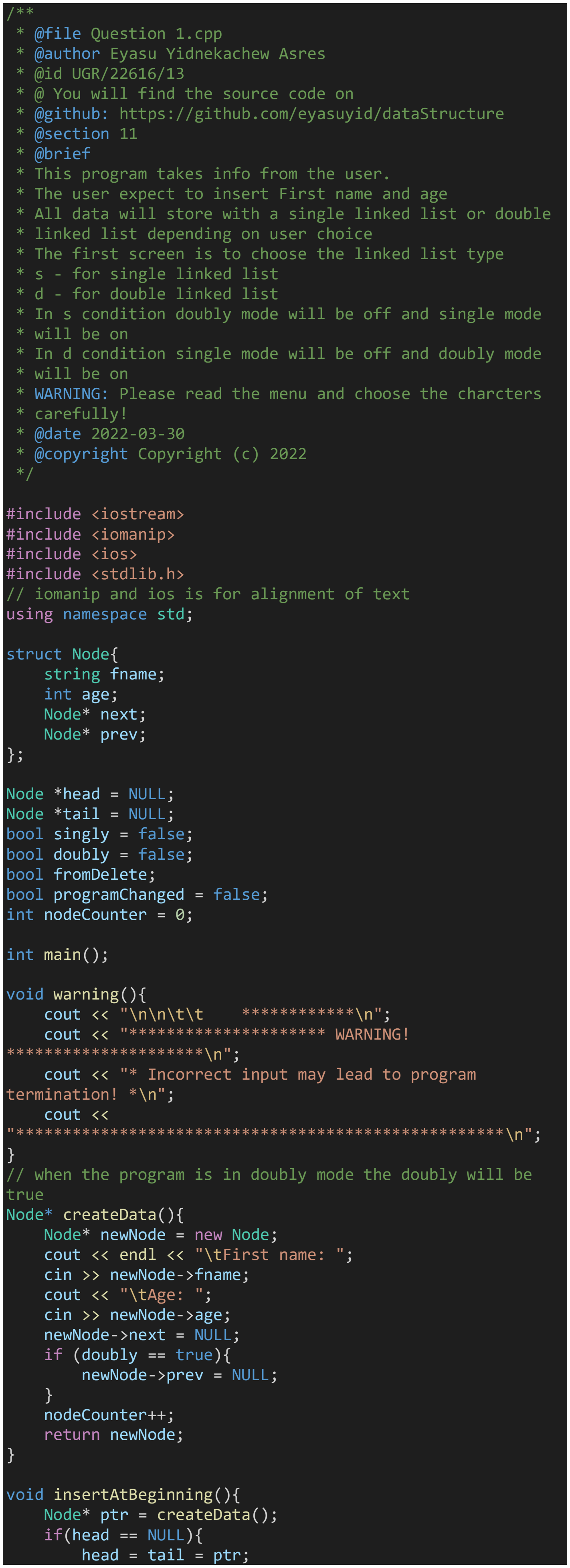
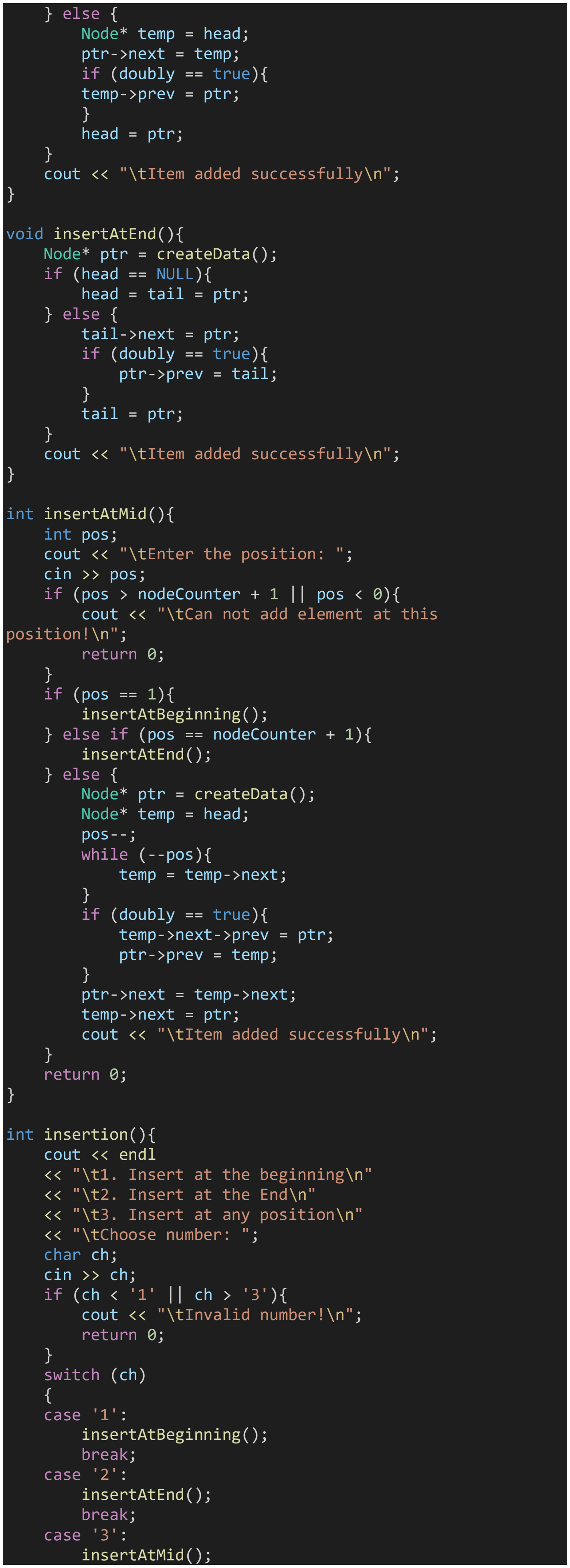
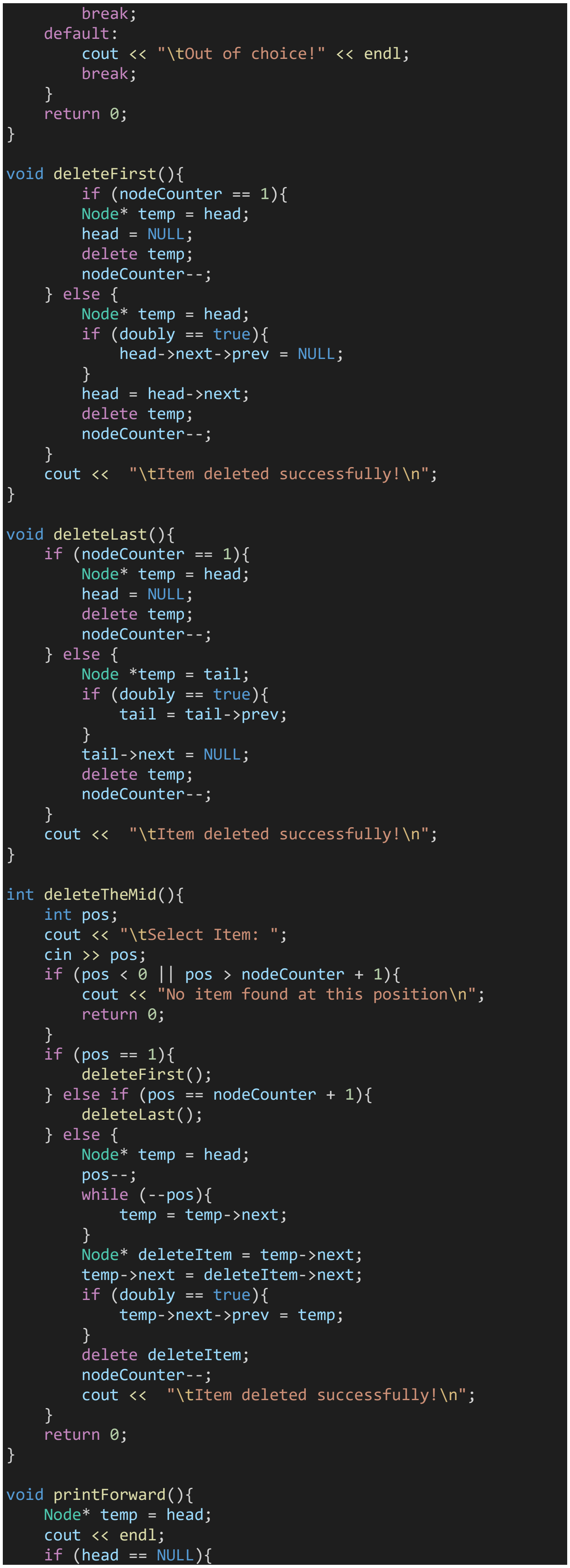


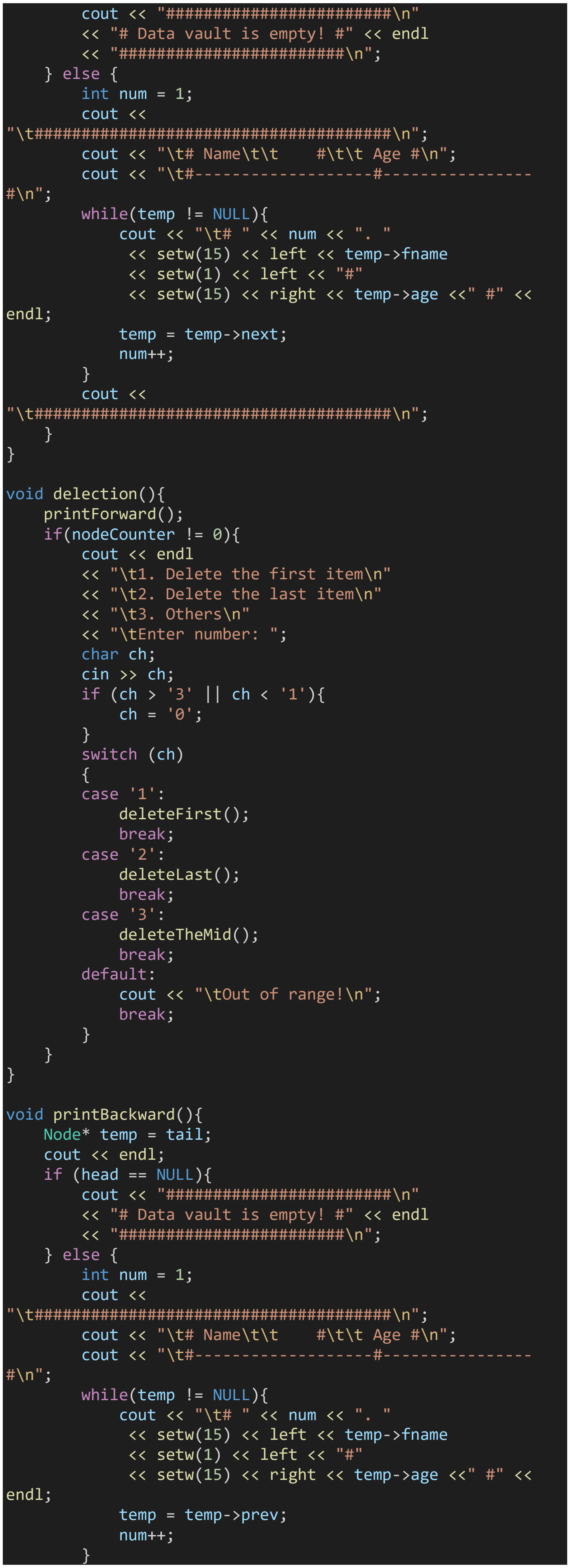
```
/**
 * @file Question 1.cpp
 * @author Eyasu Yidnekachew Asres
 * @id UGR/22616/13
 * @ You will find the source code on
 * @github: https://github.com/eyasuyid/dataStructure
 * @section 11
 * @brief
 * This program takes info from the user.
 * The user expect to insert First name and age
 * All data will store with a single linked list or double
 * linked list depending on user choice
 * The first screen is to choose the linked list type
 * s - for single linked list
 * d - for double linked list
 * In s condition doubly mode will be off and single mode
 * will be on
 * In d condition single mode will be off and doubly mode
 * will be on
 * WARNING: Please read the menu and choose the charcters
 * carefully!
 * @date 2022-03-30
 * @copyright Copyright (c) 2022
 */

#include <iostream>
#include <iomanip>
#include <ios>
#include <stdlib.h>
// iomanip and ios is for alignment of text
using namespace std;

struct Node{
    string fname;
    int age;
    Node* next;
    Node* prev;
};

Node *head = NULL;
Node *tail = NULL;
bool singly = false;
bool doubly = false;
bool fromDelete;
bool programChanged = false;
int nodeCounter = 0;

int main();

void warning(){
    cout << "\n\n\t\t    ************\n";
    cout << "********************* WARNING!
*********************\n";
    cout << "* Incorrect input may lead to program
termination! *\n";
    cout <<
"****************************************************\n";
}
// when the program is in doubly mode the doubly will be
true
Node* createData(){
    Node* newNode = new Node;
    cout << endl << "\tFirst name: ";
    cin >> newNode->fname;
    cout << "\tAge: ";
    cin >> newNode->age;
    newNode->next = NULL;
    if (doubly == true){
        newNode->prev = NULL;
    }
    nodeCounter++;
    return newNode;
}

void insertAtBeginning(){
    Node* ptr = createData();
    if(head == NULL){
        head = tail = ptr;
```



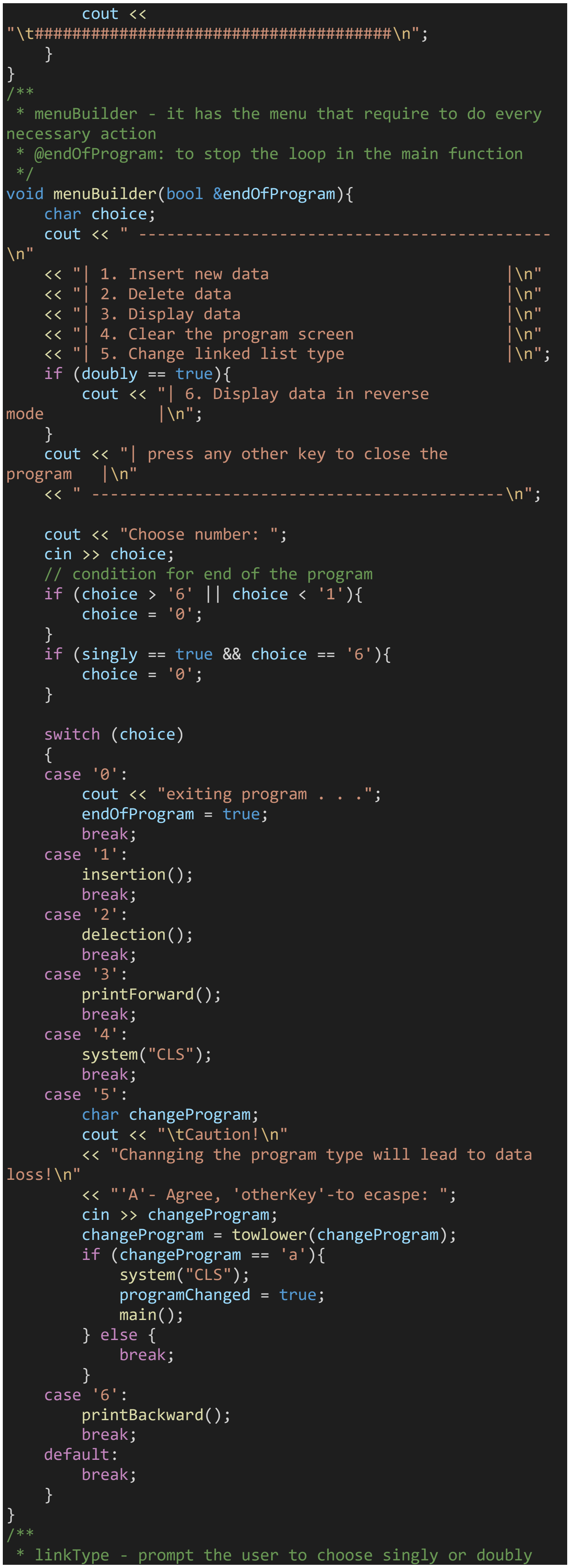
```
    } else {
        Node* temp = head;
        ptr->next = temp;
        if (doubly == true){
        temp->prev = ptr;
        }
        head = ptr;
    }
    cout << "\tItem added successfully\n";
}

void insertAtEnd(){
    Node* ptr = createData();
    if (head == NULL){
        head = tail = ptr;
    } else {
        tail->next = ptr;
        if (doubly == true){
            ptr->prev = tail;
        }
        tail = ptr;
    }
    cout << "\tItem added successfully\n";
}

int insertAtMid(){
    int pos;
    cout << "\tEnter the position: ";
    cin >> pos;
    if (pos > nodeCounter + 1 || pos < 0){
        cout << "\tCan not add element at this
position!\n";
        return 0;
    }
    if (pos == 1){
        insertAtBeginning();
    } else if (pos == nodeCounter + 1){
        insertAtEnd();
    } else {
        Node* ptr = createData();
        Node* temp = head;
        pos--;
        while (--pos){
            temp = temp->next;
        }
        if (doubly == true){
            temp->next->prev = ptr;
            ptr->prev = temp;
        }
        ptr->next = temp->next;
        temp->next = ptr;
        cout << "\tItem added successfully\n";
    }
    return 0;
}

int insertion(){
    cout << endl
    << "\t1. Insert at the beginning\n"
    << "\t2. Insert at the End\n"
    << "\t3. Insert at any position\n"
    << "\tChoose number: ";
    char ch;
    cin >> ch;
    if (ch < '1' || ch > '3'){
        cout << "\tInvalid number!\n";
        return 0;
    }
    switch (ch)
    {
    case '1':
        insertAtBeginning();
        break;
    case '2':
        insertAtEnd();
        break;
    case '3':
        insertAtMid();
```



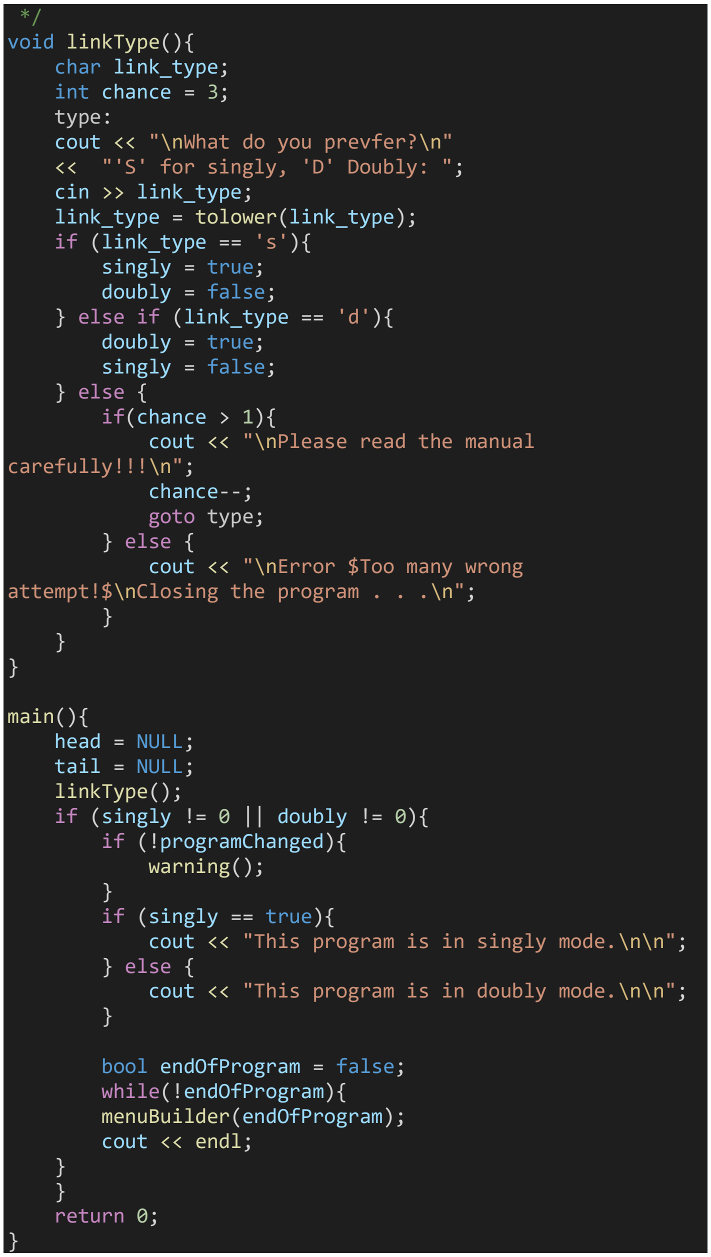
```
        break;
    default:
        cout << "\tOut of choice!" << endl;
        break;
    }
    return 0;
}

void deleteFirst(){
        if (nodeCounter == 1){
        Node* temp = head;
        head = NULL;
        delete temp;
        nodeCounter--;
    } else {
        Node* temp = head;
        if (doubly == true){
            head->next->prev = NULL;
        }
        head = head->next;
        delete temp;
        nodeCounter--;
    }
    cout <<  "\tItem deleted successfully!\n";
}

void deleteLast(){
    if (nodeCounter == 1){
        Node* temp = head;
        head = NULL;
        delete temp;
        nodeCounter--;
    } else {
        Node *temp = tail;
        if (doubly == true){
            tail = tail->prev;
        }
        tail->next = NULL;
        delete temp;
        nodeCounter--;
    }
    cout <<  "\tItem deleted successfully!\n";
}

int deleteTheMid(){
    int pos;
    cout << "\tSelect Item: ";
    cin >> pos;
    if (pos < 0 || pos > nodeCounter + 1){
        cout << "No item found at this position\n";
        return 0;
    }
    if (pos == 1){
        deleteFirst();
    } else if (pos == nodeCounter + 1){
        deleteLast();
    } else {
        Node* temp = head;
        pos--;
        while (--pos){
            temp = temp->next;
        }
        Node* deleteItem = temp->next;
        temp->next = deleteItem->next;
        if (doubly == true){
            temp->next->prev = temp;
        }
        delete deleteItem;
        nodeCounter--;
        cout <<  "\tItem deleted successfully!\n";
    }
    return 0;
}

void printForward(){
    Node* temp = head;
    cout << endl;
    if (head == NULL){
```



```
        cout << "########################\n"
        << "# Data vault is empty! #" << endl
        << "########################\n";
    } else {
        int num = 1;
        cout <<
"\t######################################\n";
        cout << "\t# Name\t\t    #\t\t Age #\n";
        cout << "\t#-------------------#----------------
#\n";
        while(temp != NULL){
            cout << "\t# " << num << ". "
             << setw(15) << left << temp->fname
             << setw(1) << left << "#"
             << setw(15) << right << temp->age <<" #" <<
endl;
            temp = temp->next;
            num++;
        }
        cout <<
"\t######################################\n";
    }
}

void delection(){
    printForward();
    if(nodeCounter != 0){
        cout << endl
        << "\t1. Delete the first item\n"
        << "\t2. Delete the last item\n"
        << "\t3. Others\n"
        << "\tEnter number: ";
        char ch;
        cin >> ch;
        if (ch > '3' || ch < '1'){
            ch = '0';
        }
        switch (ch)
        {
        case '1':
            deleteFirst();
            break;
        case '2':
            deleteLast();
            break;
        case '3':
            deleteTheMid();
            break;
        default:
            cout << "\tOut of range!\n";
            break;
        }
    }
}

void printBackward(){
    Node* temp = tail;
    cout << endl;
    if (head == NULL){
        cout << "########################\n"
        << "# Data vault is empty! #" << endl
        << "########################\n";
    } else {
        int num = 1;
        cout <<
"\t######################################\n";
        cout << "\t# Name\t\t    #\t\t Age #\n";
        cout << "\t#-------------------#----------------
#\n";
        while(temp != NULL){
            cout << "\t# " << num << ". "
             << setw(15) << left << temp->fname
             << setw(1) << left << "#"
             << setw(15) << right << temp->age <<" #" <<
endl;
            temp = temp->prev;
            num++;
        }
```



```
        cout <<
"\t######################################\n";
    }
}
/**
 * menuBuilder - it has the menu that require to do every
necessary action
 * @endOfProgram: to stop the loop in the main function
 */
void menuBuilder(bool &endOfProgram){
    char choice;
    cout << " --------------------------------------------
\n"
    << "| 1. Insert new data                          |\n"
    << "| 2. Delete data                              |\n"
    << "| 3. Display data                             |\n"
    << "| 4. Clear the program screen                 |\n"
    << "| 5. Change linked list type                  |\n";
    if (doubly == true){
        cout << "| 6. Display data in reverse
mode            |\n";
    }
    cout << "| press any other key to close the
program   |\n"
    << " --------------------------------------------\n";

    cout << "Choose number: ";
    cin >> choice;
    // condition for end of the program
    if (choice > '6' || choice < '1'){
        choice = '0';
    }
    if (singly == true && choice == '6'){
        choice = '0';
    }

    switch (choice)
    {
    case '0':
        cout << "exiting program . . .";
        endOfProgram = true;
        break;
    case '1':
        insertion();
        break;
    case '2':
        delection();
        break;
    case '3':
        printForward();
        break;
    case '4':
        system("CLS");
        break;
    case '5':
        char changeProgram;
        cout << "\tCaution!\n"
        << "Channging the program type will lead to data
loss!\n"
        << "'A'- Agree, 'otherKey'-to ecaspe: ";
        cin >> changeProgram;
        changeProgram = towlower(changeProgram);
        if (changeProgram == 'a'){
            system("CLS");
            programChanged = true;
            main();
        } else {
            break;
        }
    case '6':
        printBackward();
        break;
    default:
        break;
    }
}
/**
 * linkType - prompt the user to choose singly or doubly
```



```
 */
void linkType(){
    char link_type;
    int chance = 3;
    type:
    cout << "\nWhat do you prevfer?\n"
    <<  "'S' for singly, 'D' Doubly: ";
    cin >> link_type;
    link_type = tolower(link_type);
    if (link_type == 's'){
        singly = true;
        doubly = false;
    } else if (link_type == 'd'){
        doubly = true;
        singly = false;
    } else {
        if(chance > 1){
            cout << "\nPlease read the manual
carefully!!!\n";
            chance--;
            goto type;
        } else {
            cout << "\nError $Too many wrong
attempt!$\nClosing the program . . .\n";
        }
    }
}

main(){
    head = NULL;
    tail = NULL;
    linkType();
    if (singly != 0 || doubly != 0){
        if (!programChanged){
            warning();
        }
        if (singly == true){
            cout << "This program is in singly mode.\n\n";
        } else {
            cout << "This program is in doubly mode.\n\n";
        }

        bool endOfProgram = false;
        while(!endOfProgram){
        menuBuilder(endOfProgram);
        cout << endl;
    }
    }
    return 0;
}
```