6-08-21

(1)

```
# include <iostream>
class DB;
Class DM{
    Private:
        float meter, centimeter;
    Public:
        DM(){
            meter = 0.0;
            centimeter = 0.0;
        }
        void getdata(){
            cout << "Enter distance in m \n";
            cin >> meter;
            cout << "Enter in cm \n";
            cin >> centimeter;
        }
        void displayData(){
            cout << "The Distance is" << meter << "m,
                << centimeter << "cm\n";
        }
        void Add (DM2, DB2; int){
};
class DB{
        Private:
            float feet, inches;
        Public:
            DB(){
                feet = 0.0;
                inches = 0.0;
            }
            void getData(){
                Cout << "Enter distance inches \n";
                cin << feet;
                cout << "enter distance inches \n";
                Cin << inches;
            }
Void displayData ~~$~~ (){
    Cout << " The Distance is" << feet << " feet and"
        << inches << " inches \n";
  }
  friend void DM :: Add (DM& x, DB& y, int
                                    flag);
};

Void DM :: Add (DM& x, DB& y, int flag){
    If (flag = = 0){
        std:: Cout << " sum= " << (3.2*x.meter)
        + y. feet << " foot and << (.39*x.
          centimeter) + y.inches << " inches \n";
    }
    else {
        std::cout << " Sum= " << x.meter + y.foot/3.2
        << " meter and " << x.centimeter + y.inches/.39
        << " centimeter \n";
    }
  }

int main (){
      DM x;
      DB y;
      x. getData();
      x. displayData();
      y. get ~~D~~ata();
      y. displayData();
      int flag = 0;
      Cout << " Enter 0 for inches and 1 for meter \n"
      Cin << flag

      x. Add (x, y, flag);
      return 0;
    }
```

Q2 6/08/21

C

```
# include <iostream>
# include < bits/stdc++.h>
using namespace std;
class Node{
     Public:
       int data;
       Node* next;
};
void   delete node (Node*head, Node* n){
       if(need ==n)
       {
          if (head -> next == NULL){
              Cout<< "There is onlyone Node";
              return
          }
                                        data
          head -> data = head -> next -> ~~next~~
            n= head -> next;
           head -> next = head -> next->next;
           free(n);
           return;
       }
      Node* prev = head;
      While (prev -> next != NULL && prev -> next
                 !=n) {
           Prev = prev -> next;
       }.
       ~~if(prev -> next == NULL){~~
            ~~cout<< "Node not present"~~;
          ~~return;~~
       Prev => next = prev -> next -> next;
       free (n) ;
       return;
    }
```

```
void push(Node **head_ref, int ~~[illegible]~~ data){
        Node *new_node = new Node();
        new - node -> data = new data;
        new - node -> next = *head_ref;
        *head-ref = new -node;
        }

void print list (Node * head){
        While (head != NULL){
            Cout << head -> data << " ";
            head = head -> next;
        }
    }
int   main (){
        Node * head = NULL;
        Char C;
        cout << " Press Y to enter data";
        cin >> C;
        While (c=='y' || C=='Y'):
            int data;
            Cout << "enter data";
            cin >> data ;
             push (&head, data);
             cout << " press Y to enter more";
             Cin >> C;
        }
       Print list(head);
       cout << "enter node to delete";
       Cin >> data
       Node* temp= head;
        While (temp -> data != data){
              temp = temp -> next;
        }
        If (temp ~~[illegible]~~ != NULL){
            delete node (head, temp);
        }
        else{
```

```
        cout >> "Node not there";
    }
    printlist(head);
    return0;
}
```