#include <bits/stdc++.h>

using namespace std;

class node

{

public:

int rollno;

node \*next;

node(int num) //constructor

{

rollno = num;

next = NULL;

}

}\*all\_head = NULL, \*vanilla\_head = NULL, \*butter\_head = NULL, \*union\_head = NULL, \*inters\_head = NULL, \*diff\_head = NULL;

void append(node \*\*head\_ref) //insert at end

{

int num;

cout<<"Enter Rollno: ";

cin>>num;

node \*new\_node = new node(num);

if (\*head\_ref == NULL) //if list is empty

{

\*head\_ref = new\_node;

return;

}

node \*last = \*head\_ref;

while (last -> next != NULL)

{

last = last -> next;

}

last -> next = new\_node;

}

void append(node \*\*head\_ref, int key)

{

node \*new\_node = new node(key);

if (\*head\_ref == NULL) //if list is empty

{

\*head\_ref = new\_node;

return;

}

node \*last = \*head\_ref;

while (last -> next != NULL)

{

last = last -> next;

}

last -> next = new\_node;

}

void display\_list(node \*\*head\_ref)

{

cout<<"\n";

node \*temp = \*head\_ref;

if (temp == NULL)

{

cout<<"List is empty";

return;

}

while (temp != NULL)

{

cout<<temp -> rollno<<" ";

temp = temp -> next;

}

cout<<"\n";

}

void get\_all()

{

int count;

cout<<"\nEnter total students: ";

cin>>count;

for (int i = 0; i < count; i++)

{

append(&all\_head);

}

cout<<"\nAll students: ";

display\_list(&all\_head);

}

void get\_vanilla()

{

int count;

cout<<"\nEnter total students who like vanilla: ";

cin>>count;

for (int i = 0; i < count; i++)

{

append(&vanilla\_head);

}

cout<<"\nStudents who like vanilla: ";

display\_list(&vanilla\_head);

}

void get\_butter()

{

int count;

cout<<"\nEnter total students who like butterscotch: ";

cin>>count;

for (int i = 0; i < count; i++)

{

append(&butter\_head);

}

cout<<"\nStudents who like butterscotch: ";

display\_list(&butter\_head);

}

node \*\*unio(node \*\*head1, node \*\*head2)

{

//fill union list with all elements of list 1

node \*temp1 = \*head1;

while (temp1 != NULL)

{

append(&union\_head, (temp1 -> rollno));

temp1 = temp1 -> next;

}

//fill union list with non-common elements from list 2

node \*temp2 = \*head2;

while (temp2 != NULL)

{

temp1 = \*head1;

bool flag = false;

while (temp1 != NULL)

{

if ((temp2 -> rollno) == (temp1 -> rollno))

flag = true;

temp1 = temp1 -> next;

}

if (!flag)

{

append(&union\_head, (temp2 -> rollno));

}

temp2 = temp2 -> next;

}

return &union\_head;

}

node \*\*inters(node \*\*head1, node \*\*head2)

{

node \*temp1 = \*head1;

while (temp1 != NULL)

{

node \*temp2 = \*head2;

while (temp2 != NULL)

{

if ((temp1 -> rollno) == (temp2 -> rollno))

{

append(&inters\_head, (temp1 -> rollno));

}

temp2 = temp2 -> next;

}

temp1 = temp1 -> next;

}

return &inters\_head;

}

node \*\*diff(node \*\*head1, node \*\*head2)

{

node \*temp1 = \*head1;

while (temp1 != NULL)

{

node \*temp2 = \*head2;

bool flag = false;

while (temp2 != NULL)

{

if ((temp1 -> rollno) == (temp2 -> rollno))

{

flag = true;

}

temp2 = temp2 -> next;

}

if(!flag)

{

append(&diff\_head, (temp1 -> rollno));

}

temp1 = temp1 -> next;

}

return &diff\_head;

}

int main()

{

bool exit = false;

int choice;

do

{

cout<<"\n1. To enter data";

cout<<"\n2. To display the rollnos of students who like vanila or butterscotch";

cout<<"\n3. To display the rollnos of students who like only vanila";

cout<<"\n4. To display the rollnos of students who like only butterscotch";

cout<<"\n5. To display the rollnos of students who like both vanila and butterscotch ";

cout<<"\n6. To display the rollnos of students who neither like vanila nor butterscotch";

cout<<"\n7. Exit";

cout<<"\nEnter choice: ";

cin>>choice;

switch (choice)

{

case 1:

get\_all();

get\_vanilla();

get\_butter();

break;

case 2:

cout<<"Students who like vanilla or butterscotch: ";

display\_list( unio(&vanilla\_head, &butter\_head) );

break;

case 3:

cout<<"Students who like only vanilla: ";

display\_list(&vanilla\_head);

break;

case 4:

cout<<"Students who like butterscotch: ";

display\_list(&butter\_head);

break;

case 5:

cout<<"Students who like both vanilla and butterscotch: ";

display\_list( inters(&vanilla\_head, &butter\_head) );

break;

case 6:

cout<<"Students who like neither vanilla nor butterscotch: ";

display\_list( diff(&all\_head, unio(&vanilla\_head, &butter\_head)) );

break;

case 7:

exit = true;

break;

}

} while (!exit);

}