#include<iostream>

using namespace std;

struct emp{

int empid,salary;

string name,mob,design;

emp \*link;

};

struct emp \*create\_record(){

emp \*emp1;

emp1=new emp;

cout<<"ENTER THE EMPLOYEE ID:";

cin>>emp1->empid;

cout<<"ENTER THE NAME OF EMPLOYEE:";

cin>>emp1->name;

cout<<"ENTER THE DESIGNATION OF EMPLOYEE:";

cin>>emp1->design;

cout<<"ENTER THE MOBILE NO. OF EMPLOYEE:";

cin>>emp1->mob;

cout<<"ENTER THE SALARY OF EMPLOYEE:";

cin>>emp1->salary;

emp1->link=NULL;

return emp1;

}

struct emp\* create\_list(){

emp \*emp1,\*head=NULL,\*ptr;

char ch;

do{

emp1=create\_record();

if(emp1==NULL){

cout<<"MEMORY IS NOT ALLOCATED\n";

}

else{

if(head==NULL)

head=emp1;

else{

ptr=head;

while(ptr->link!=NULL){

ptr=ptr->link;

}

ptr->link=emp1;

}

cout<<"WANT TO ENTER MORE RECORDS(Y/N):";

cin>>ch;

}

}while(ch=='y' || ch=='Y');

return head;

}

void display(emp \*head){

emp \*emp1;

if(head==NULL){

cout<<"NO RECORD IS PRESENT\n";

}

else{

emp1=head;

while(emp1!=NULL){

cout<<"EMP ID:"<<emp1->empid<<"->";

cout<<"EMP NAME:"<<emp1->name<<"->";

cout<<"EMP DESIGNATION:"<<emp1->design<<"->";

cout<<"EMP MOBILE NO.:"<<emp1->mob<<"->";

cout<<"EMP SALARY:"<<emp1->salary<<"->";

emp1=emp1->link;

cout<<"\n";

}

cout<<"END\n";

}

}

struct emp\* insert\_front(emp \*head){

emp \*emp1,\*ptr;

emp1=create\_record();

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

ptr=head;

emp1->link=ptr;

head=emp1;

}

return head;

}

void insert\_middle(emp\* head)

{

emp \*emp1,\*ptr;

int key,f=0;

emp1=create\_record();

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

ptr=head;

cout<<"ENTER THE EMP ID AFTER WHICH NEW RECORD WILL INSERT:";

cin>>key;

while(ptr!=NULL){

if(ptr->empid==key){

emp1->link=ptr->link;

ptr->link=emp1;

f=1;

}

ptr=ptr->link;

}

if(f==0){

cout<<"RECORD IS NOT FOUND AFTER WHICH NEW RECORD WILL INSERT\n";

}

else{

cout<<"SUCCESSFULLY RECORD IS INSERTED\n";

}

}

}

void insert\_end(emp \*head){

emp \*emp1,\*ptr;

emp1=create\_record();

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

ptr=head;

while(ptr->link!=NULL){

ptr=ptr->link;

}

ptr->link=emp1;

}

}

struct emp\* delete\_front(emp\* head){

emp \*ptr1,\*ptr2;

int f=0;

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

ptr1=head;

ptr2=ptr1->link;

head=ptr2;

delete ptr1;

cout<<"RECORD AT FRONT IS SUCCESSFULLY DELETED\n";

}

return head;

}

void delete\_middle(emp\* head){

emp \*ptr1,\*ptr2;

int key,f=0;

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

cout<<"ENTER THE EMP ID FOR DELETION OF RECORD:";

cin>>key;

ptr1=head;

while(ptr1->link!=NULL){

if(ptr1->empid==key){

ptr2->link=ptr1->link;

delete ptr1;

f=1;

break;

}

ptr2=ptr1;

ptr1=ptr1->link;

}

if(f==0){

cout<<"THE RECORD FOR DELETION IS NOT PRESENT\n";

}

else{

cout<<"RECORD IS SUCCESSFULLY DELETED\n";

}

}

}

void delete\_end(emp\* head){

emp \*ptr1,\*ptr2;

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

ptr1=head;

while(ptr1->link!=NULL){

ptr2=ptr1;

ptr1=ptr1->link;

}

ptr2->link=NULL;

delete ptr1;

cout<<"RECORD SUCCESSFULLY DELETED\n";

}

}

void search(emp\* head){

emp \*ptr;

int key,f=0;

if(head==NULL){

cout<<"LIST IS EMPTY\n";

}

else{

cout<<"ENTER THE EMP ID FOR SEARCHING A RECORD:";

cin>>key;

ptr=head;

while(ptr!=NULL){

if(ptr->empid==key){

cout<<"EMP ID:"<<ptr->empid<<"->";

cout<<"EMP NAME:"<<ptr->name<<"->";

cout<<"EMP DESIGNATION:"<<ptr->design<<"->";

cout<<"EMP MOBILE NO.:"<<ptr->mob<<"->";

cout<<"EMP SALARY:"<<ptr->salary<<endl;

f=1;

}

ptr=ptr->link;

}

if(f==0){

cout<<"RECORD IS NOT FOUND\n";

}

}

}

int main()

{

emp \*emp1;

int ch;

char f;

cout<<"\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\n";

cout<<"1.MAKE LIST\n2.INSERT AT FRONT\n3.INSERT AT MIDDLE\n4.INSERT AT END\n5.DELETE AT FRONT\n6.DELETE AT MIDDLE\n7.DELETE AT END\n8.SERACH\n9.DISPLAY\n10.EXIT\n";

do{

cout<<"ENTER YOUR CHOICE:";

cin>>ch;

switch(ch){

case 1:

emp1=create\_list();

break;

case 2:

emp1=insert\_front(emp1);

break;

case 3:

insert\_middle(emp1);

break;

case 4:

insert\_end(emp1);

break;

case 5:

emp1=delete\_front(emp1);

break;

case 6:

delete\_middle(emp1);

break;

case 7:

delete\_end(emp1);

break;

case 8:

search(emp1);

break;

case 9:

display(emp1);

break;

case 10:

cout<<"END OF THE PROGRAM\n";

exit(0);

break;

}

cout<<"WANT TO CONTINUE(Y/N):";

cin>>f;

}while(f=='Y' || f=='y');

return 0;

}