# LAPORAN TUGAS BESAR

# KEGIATAN TALENT SEBUAH AGENCY

**Mata Kuliah : Struktur Data**

**Kelompok : 9**

**Nama Anggota :**

**Agil Ditya Rafiazmi (1301223044)**

**Rafianto Tri Aushaf (1301223274)**

**PROGRAM STUDI S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY**

## ABSTRAK

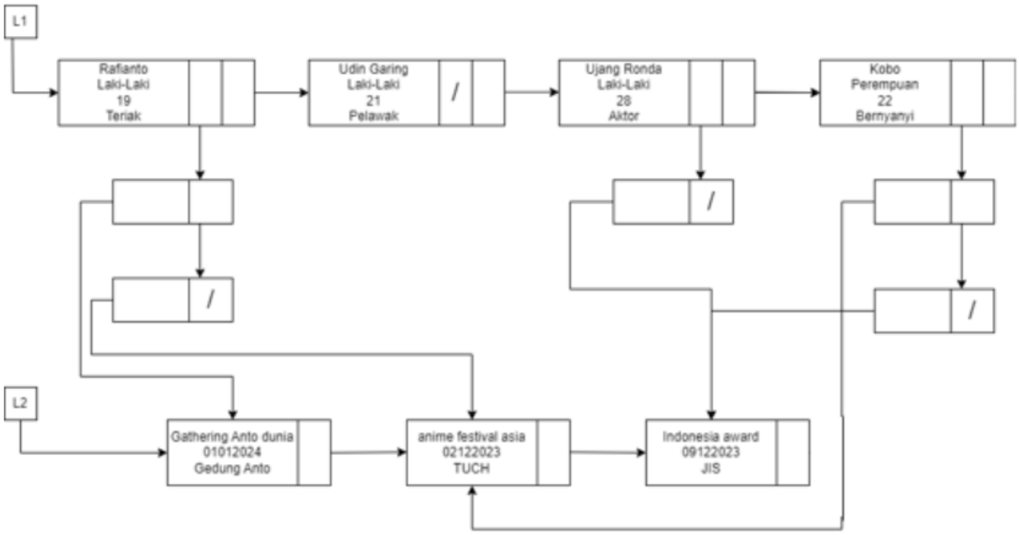
Laporan ini membahas tugas besar struktur data dengan fokus materi multi linked list. Topik yang diambil oleh kelompok kami adalah Talent dari sebuah agency yang memiliki Acara. Secara fungsi program yang kami buat dapat menyimpan data acara dari beberapa talent yang ada. Fokus materi disini adalah multi linked list dengan relasi M to N. List induk atau parents akan menyimpan informasi Talent  yang berisi Nama, jenis kelamin, umur dan bakat Talent. Pada list child akan menyimpan informasi Acara yang berisi Nama Acara, tanggal dan tempat acara.

## PENDAHULUAN

Laporan ini membahas code yang kami gunakan untuk menjalankan program ini. kami juga memberikan beberapa asumsi untuk menghindari code error seperti :

- Tidak ada talent yang duplikat
- Tidak ada acara duplikat
- Tidak ada relasi duplikat

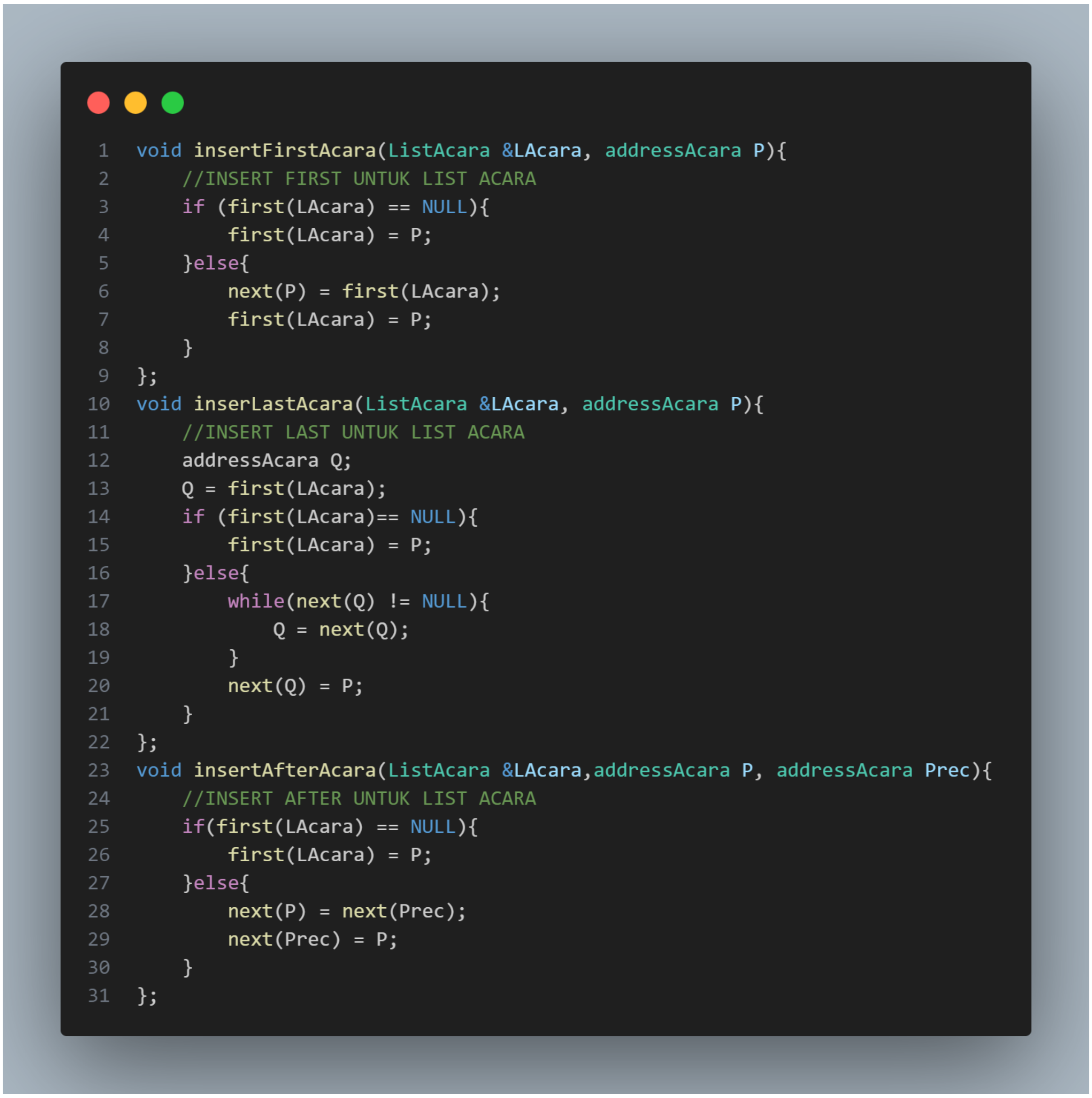
## VISUALISASI



## Fungsionalitas

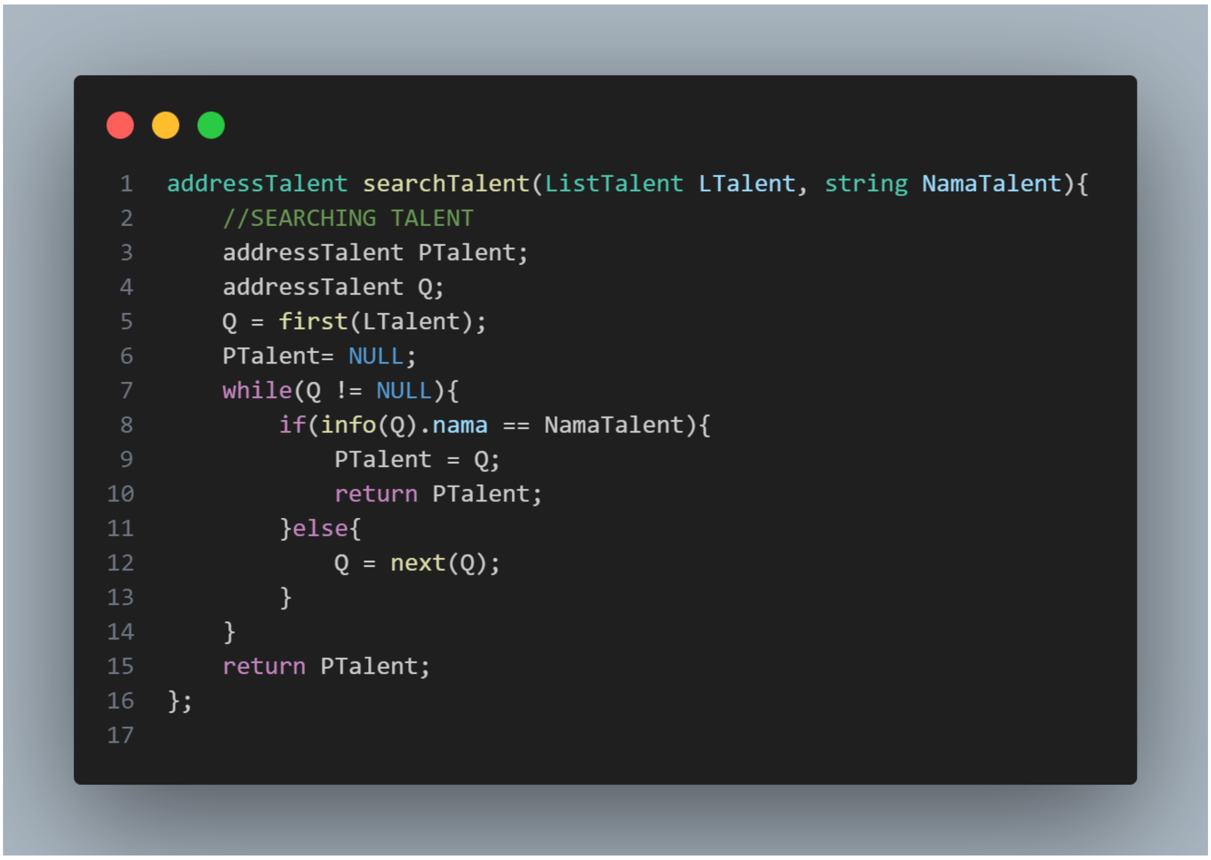
//Parents
1. Insert Talent (Insert First/Insert After/Insert Last)
2. Search Talent
3. Delete Talent (Dan Relasinya) (Delete First/Delete After/Delete Last)
4. show all Talent
//child
5. Insert  Acara (Insert First/Insert After/Insert Last)
6. Search Acara
7. Delete Acara (Delete First/Delete After/Delete Last)
8. Show All Acara
// Relasi Parent Child
9. Hubungkan Relasi Talent dan Acara (Insert First/Insert After/Insert Last)
10. Search Acara pada Talent
11. Delete Acara pada Talent (Delete First/Delete After/Delete Last)
12. Count Acara pada Talent
13. Show semua Talent dan acara yang ada
14. Show Acara yang dimiliki Talent
15. Show Semua Relasi
0. Keluar Program

1. Insert Talent (Insert First/Insert After/Insert Last)

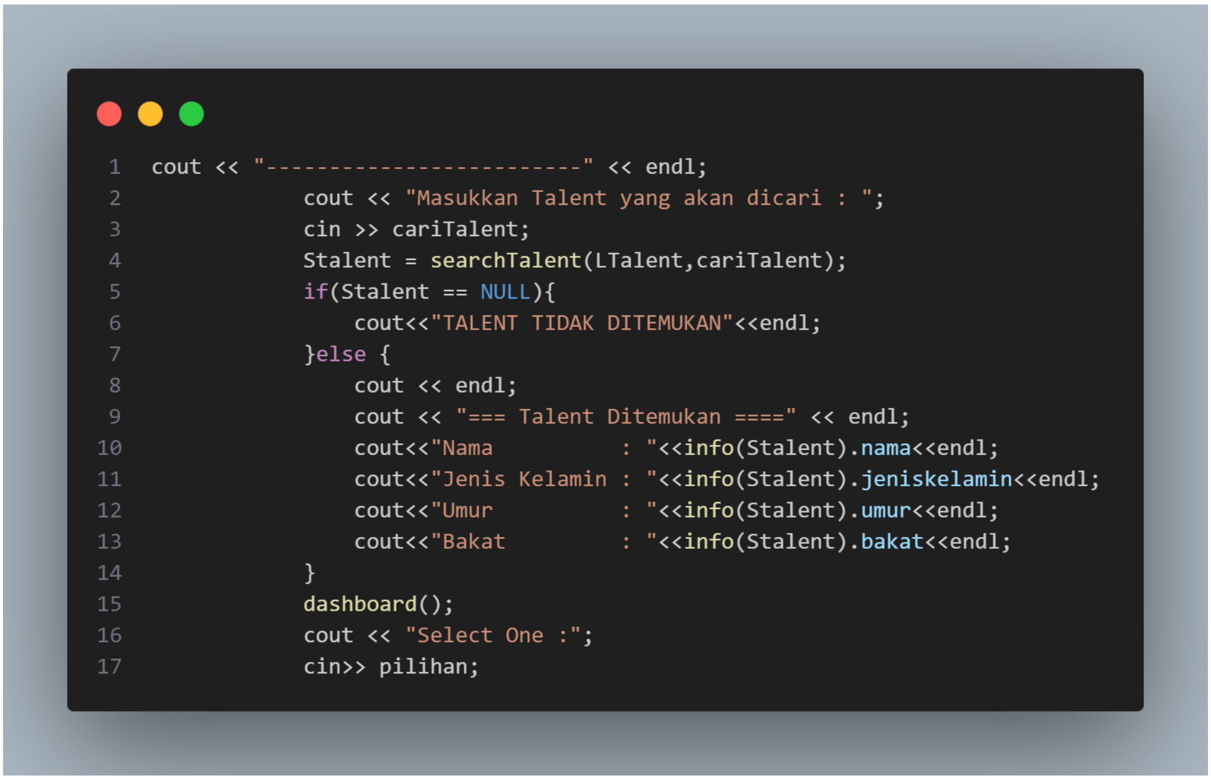


```
void insertFirstAcara(ListAcara &LAcara, addressAcara P){
    //INSERT FIRST UNTUK LIST ACARA
    if (first(LAcara) == NULL){
        first(LAcara) = P;
    }else{
        next(P) = first(LAcara);
        first(LAcara) = P;
    }
};
void inserLastAcara(ListAcara &LAcara, addressAcara P){
    //INSERT LAST UNTUK LIST ACARA
    addressAcara Q;
    Q = first(LAcara);
    if (first(LAcara)== NULL){
        first(LAcara) = P;
    }else{
        while(next(Q) != NULL){
            Q = next(Q);
        }
        next(Q) = P;
    }
};
void insertAfterAcara(ListAcara &LAcara,addressAcara P, addressAcara Prec){
    //INSERT AFTER UNTUK LIST ACARA
    if(first(LAcara) == NULL){
        first(LAcara) = P;
    }else{
        next(P) = next(Prec);
        next(Prec) = P;
    }
};
```

2. Search Talent

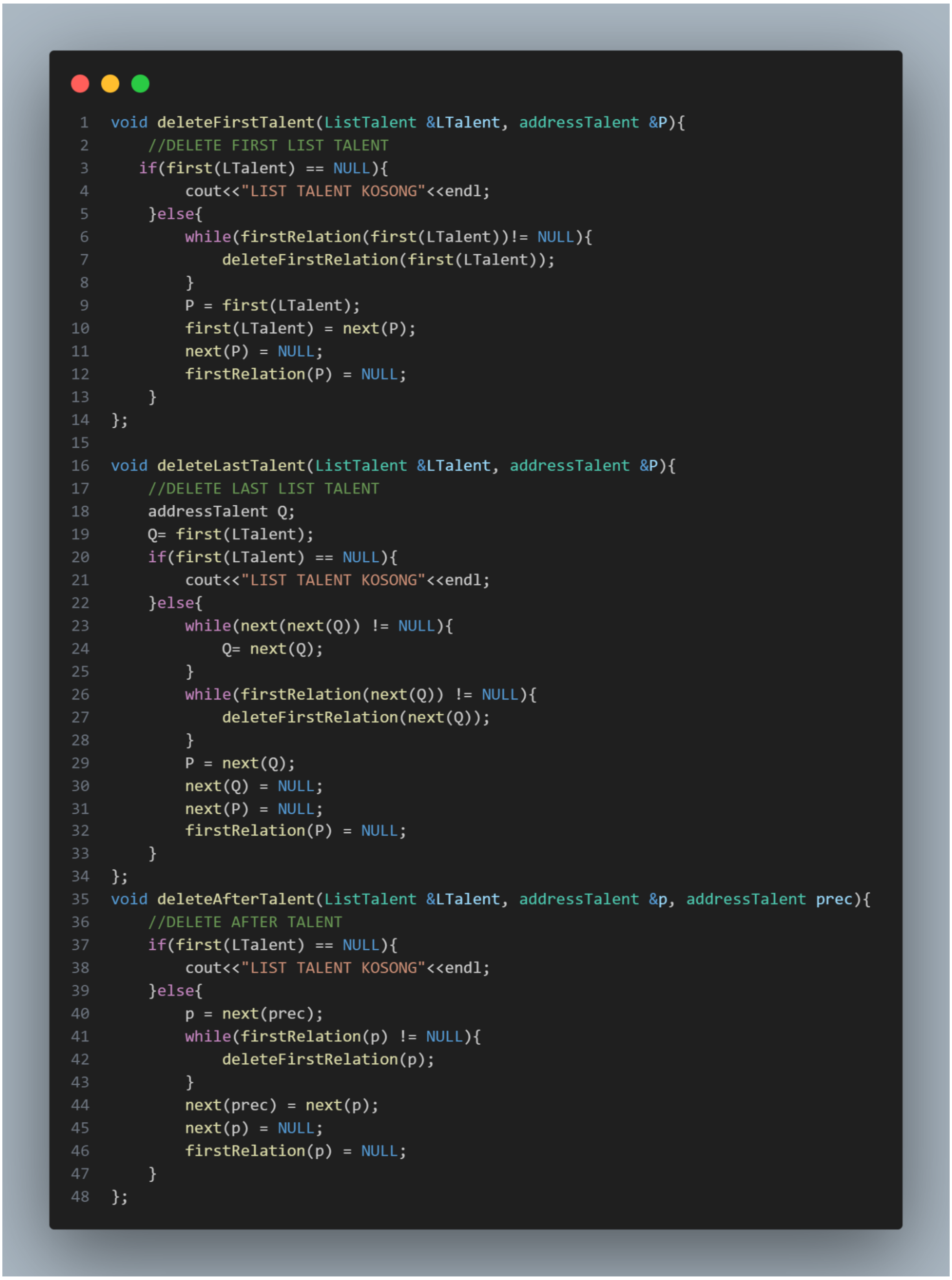


```
addressTalent searchTalent(ListTalent LTalent, string NamaTalent){
    //SEARCHING TALENT
    addressTalent PTalent;
    addressTalent Q;
    Q = first(LTalent);
    PTalent= NULL;
    while(Q != NULL){
        if(info(Q).nama == NamaTalent){
            PTalent = Q;
            return PTalent;
        }else{
            Q = next(Q);
        }
    }
    return PTalent;
};
```



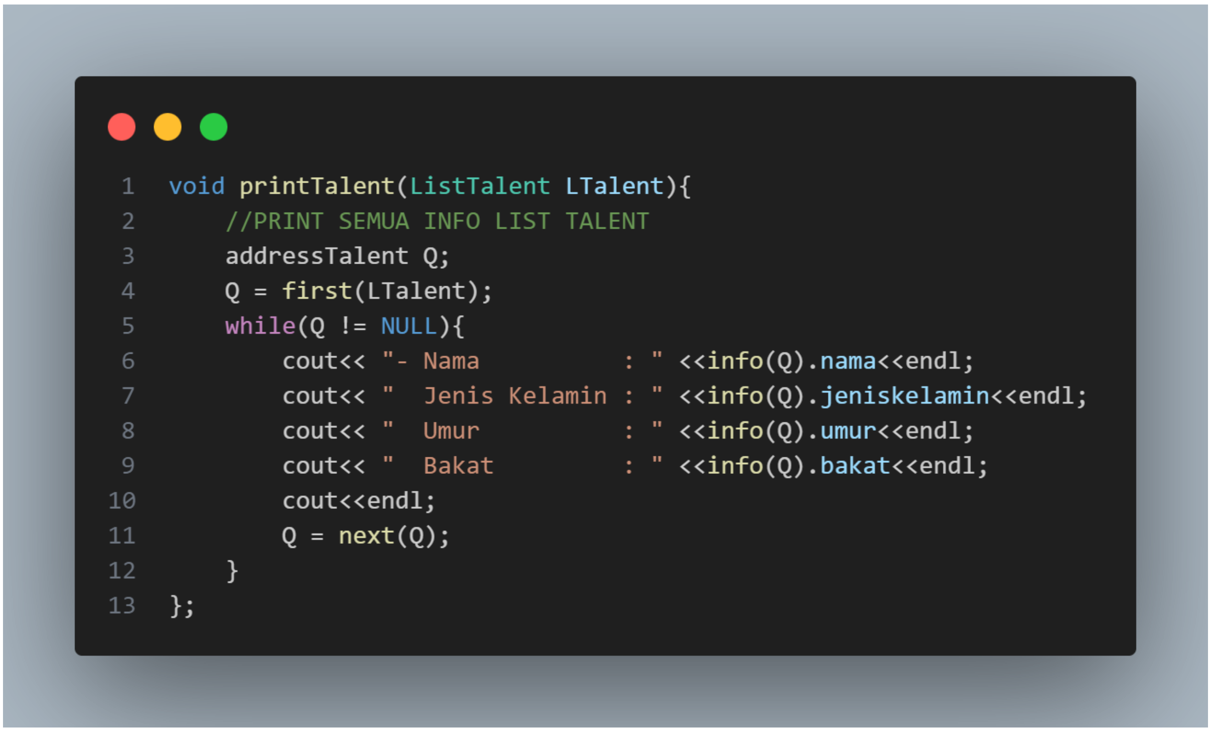
```
cout << "-------------------------" << endl;
            cout << "Masukkan Talent yang akan dicari : ";
            cin >> cariTalent;
            Stalent = searchTalent(LTalent,cariTalent);
            if(Stalent == NULL){
                cout<<"TALENT TIDAK DITEMUKAN"<<endl;
            }else {
                cout << endl;
                cout << "=== Talent Ditemukan ====" << endl;
                cout<<"Nama          : "<<info(Stalent).nama<<endl;
                cout<<"Jenis Kelamin : "<<info(Stalent).jeniskelamin<<endl;
                cout<<"Umur          : "<<info(Stalent).umur<<endl;
                cout<<"Bakat         : "<<info(Stalent).bakat<<endl;
            }
            dashboard();
            cout << "Select One :";
            cin>> pilihan;
```

3. Delete Talent (Dan Relasinya) (Delete First/Delete After/Delete Last)



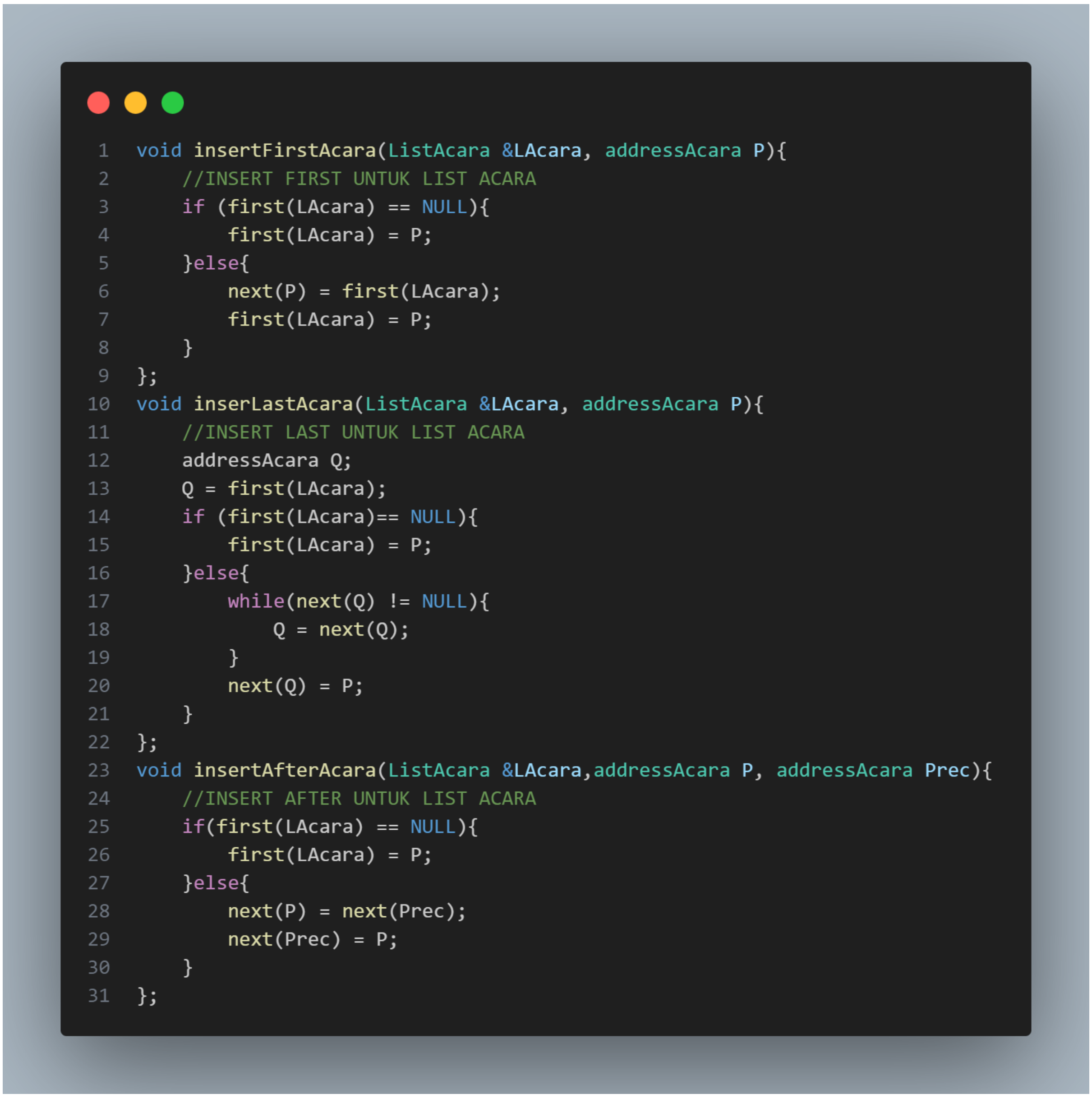
```
void deleteFirstTalent(ListTalent &LTalent, addressTalent &P){
    //DELETE FIRST LIST TALENT
   if(first(LTalent) == NULL){
        cout<<"LIST TALENT KOSONG"<<endl;
    }else{
        while(firstRelation(first(LTalent))!= NULL){
            deleteFirstRelation(first(LTalent));
        }
        P = first(LTalent);
        first(LTalent) = next(P);
        next(P) = NULL;
        firstRelation(P) = NULL;
    }
};

void deleteLastTalent(ListTalent &LTalent, addressTalent &P){
    //DELETE LAST LIST TALENT
    addressTalent Q;
    Q= first(LTalent);
    if(first(LTalent) == NULL){
        cout<<"LIST TALENT KOSONG"<<endl;
    }else{
        while(next(next(Q)) != NULL){
            Q= next(Q);
        }
        while(firstRelation(next(Q)) != NULL){
            deleteFirstRelation(next(Q));
        }
        P = next(Q);
        next(Q) = NULL;
        next(P) = NULL;
        firstRelation(P) = NULL;
    }
};
void deleteAfterTalent(ListTalent &LTalent, addressTalent &p, addressTalent prec){
    //DELETE AFTER TALENT
    if(first(LTalent) == NULL){
        cout<<"LIST TALENT KOSONG"<<endl;
    }else{
        p = next(prec);
        while(firstRelation(p) != NULL){
            deleteFirstRelation(p);
        }
        next(prec) = next(p);
        next(p) = NULL;
        firstRelation(p) = NULL;
    }
};
```

4. show all Talent



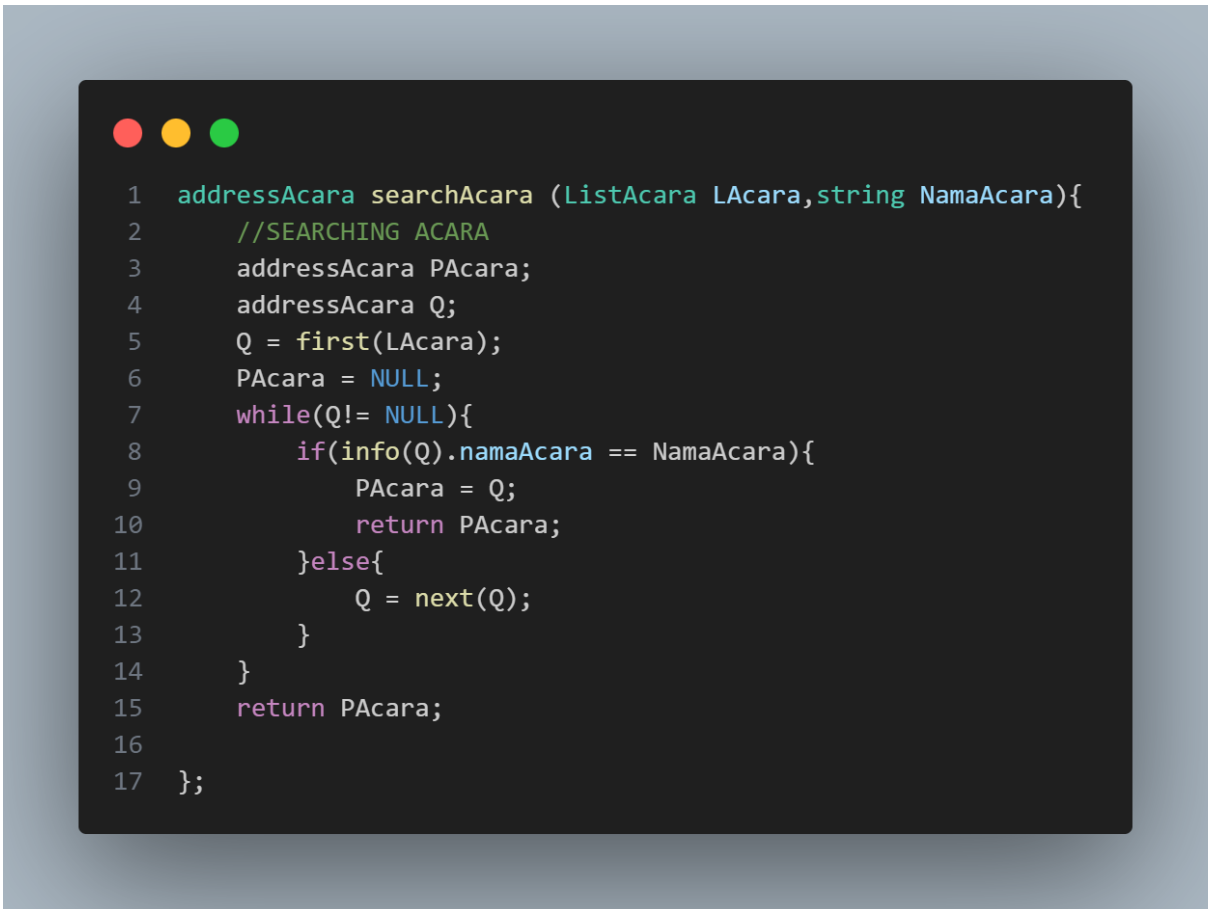
```
void printTalent(ListTalent LTalent){
    //PRINT SEMUA INFO LIST TALENT
    addressTalent Q;
    Q = first(LTalent);
    while(Q != NULL){
        cout<< "- Nama          : " <<info(Q).nama<<endl;
        cout<< "  Jenis Kelamin : " <<info(Q).jeniskelamin<<endl;
        cout<< "  Umur          : " <<info(Q).umur<<endl;
        cout<< "  Bakat         : " <<info(Q).bakat<<endl;
        cout<<endl;
        Q = next(Q);
    }
};
```

5. Insert  Acara (Insert First/Insert After/Insert Last)

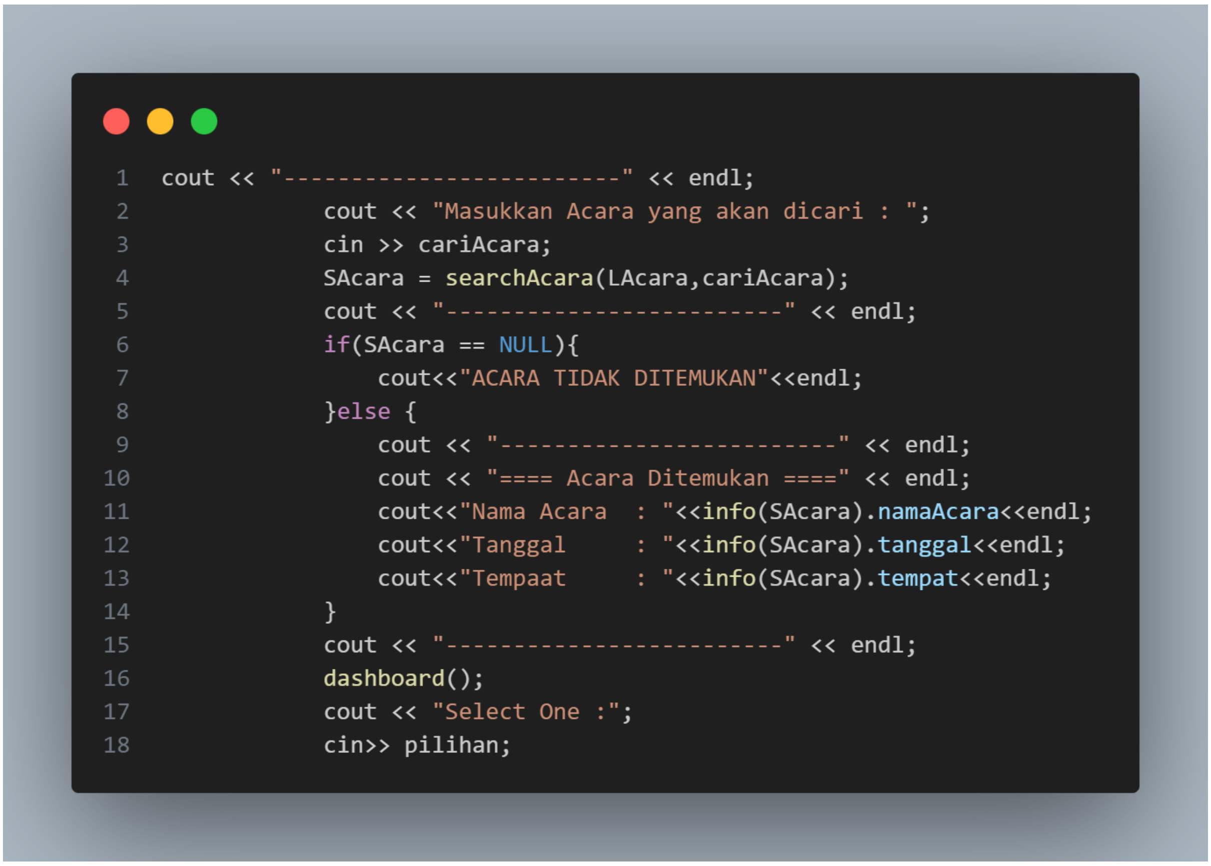


```
void insertFirstAcara(ListAcara &LAcara, addressAcara P){
    //INSERT FIRST UNTUK LIST ACARA
    if (first(LAcara) == NULL){
        first(LAcara) = P;
    }else{
        next(P) = first(LAcara);
        first(LAcara) = P;
    }
};
void inserLastAcara(ListAcara &LAcara, addressAcara P){
    //INSERT LAST UNTUK LIST ACARA
    addressAcara Q;
    Q = first(LAcara);
    if (first(LAcara)== NULL){
        first(LAcara) = P;
    }else{
        while(next(Q) != NULL){
            Q = next(Q);
        }
        next(Q) = P;
    }
};
void insertAfterAcara(ListAcara &LAcara,addressAcara P, addressAcara Prec){
    //INSERT AFTER UNTUK LIST ACARA
    if(first(LAcara) == NULL){
        first(LAcara) = P;
    }else{
        next(P) = next(Prec);
        next(Prec) = P;
    }
};
```

6. Search Acara

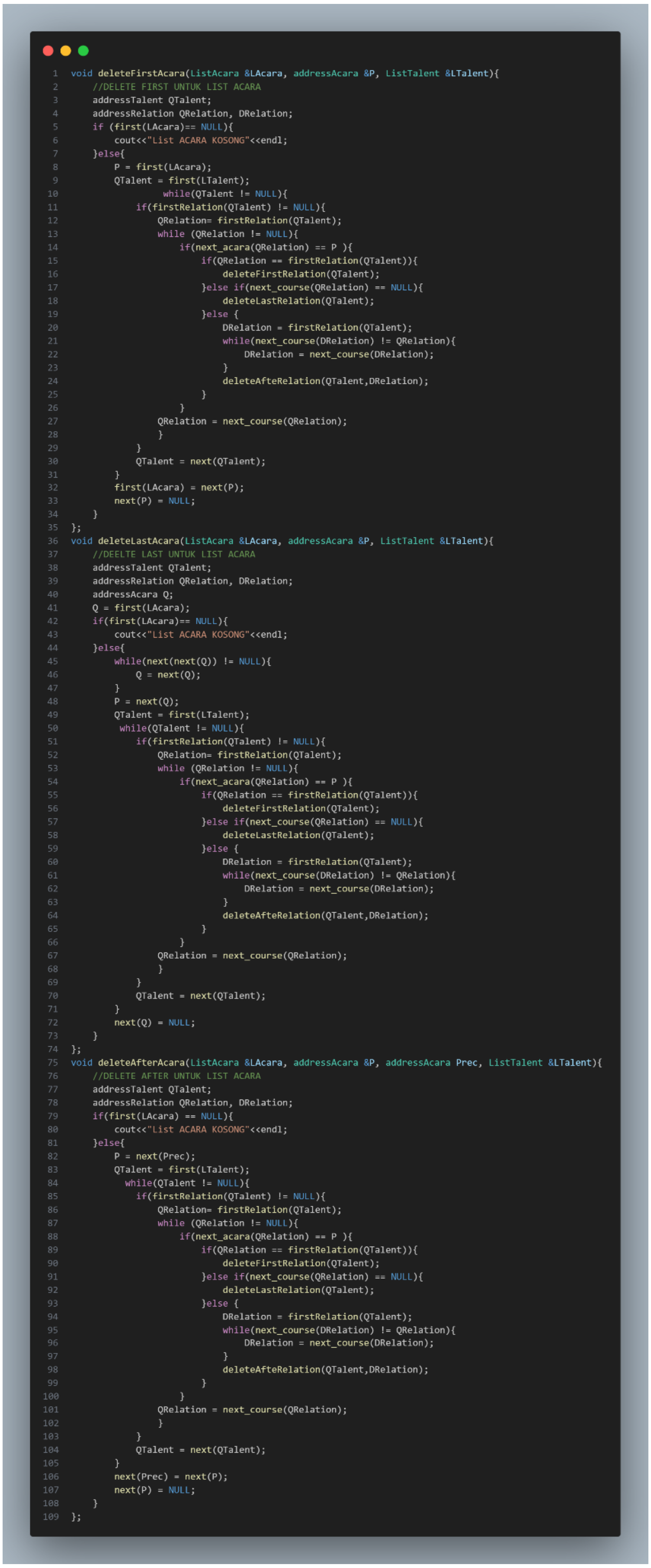


```cpp
addressAcara searchAcara (ListAcara LAcara,string NamaAcara){
    //SEARCHING ACARA
    addressAcara PAcara;
    addressAcara Q;
    Q = first(LAcara);
    PAcara = NULL;
    while(Q!= NULL){
        if(info(Q).namaAcara == NamaAcara){
            PAcara = Q;
            return PAcara;
        }else{
            Q = next(Q);
        }
    }
    return PAcara;

};
```



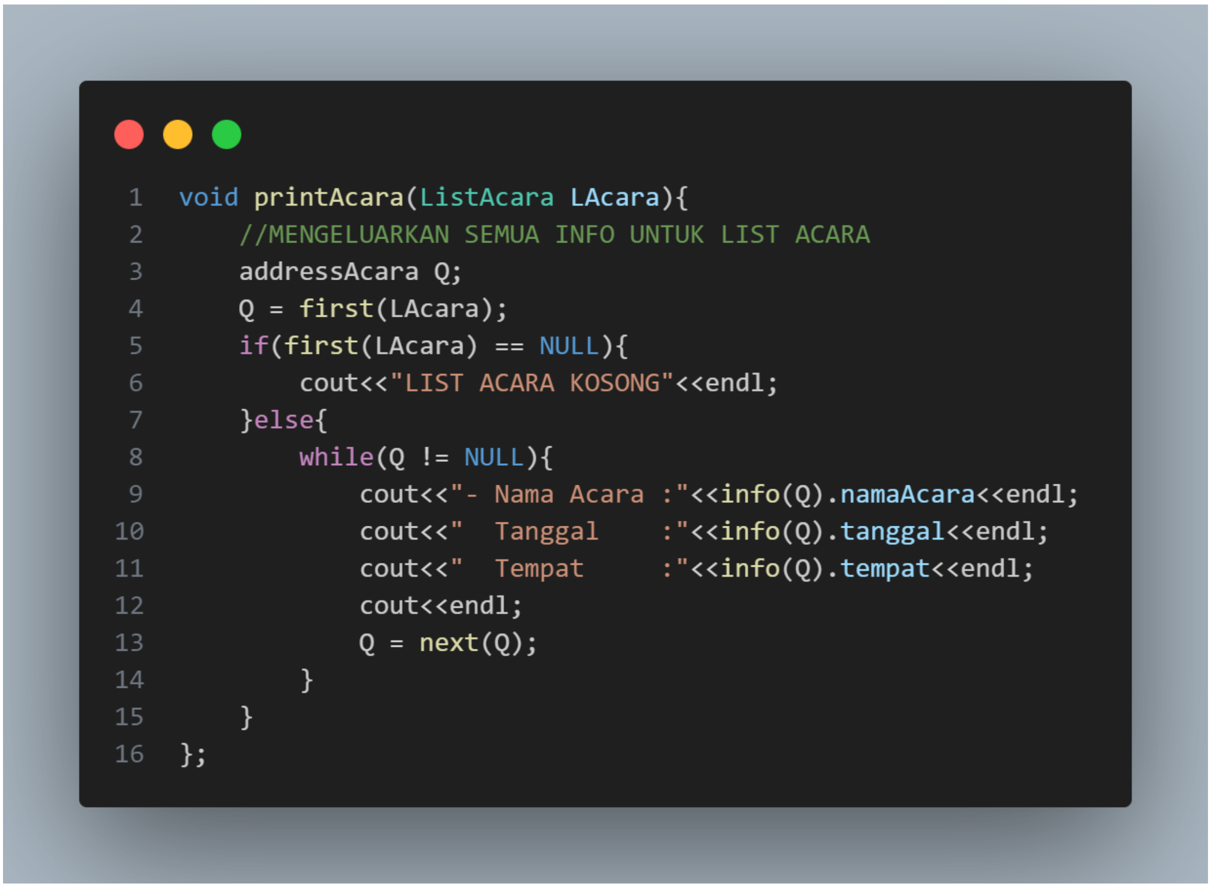
```cpp
cout << "-------------------------" << endl;
            cout << "Masukkan Acara yang akan dicari : ";
            cin >> cariAcara;
            SAcara = searchAcara(LAcara,cariAcara);
            cout << "-------------------------" << endl;
            if(SAcara == NULL){
                cout<<"ACARA TIDAK DITEMUKAN"<<endl;
            }else {
                cout << "-------------------------" << endl;
                cout << "==== Acara Ditemukan ====" << endl;
                cout<<"Nama Acara  : "<<info(SAcara).namaAcara<<endl;
                cout<<"Tanggal     : "<<info(SAcara).tanggal<<endl;
                cout<<"Tempaat     : "<<info(SAcara).tempat<<endl;
            }
            cout << "-------------------------" << endl;
            dashboard();
            cout << "Select One :";
            cin>> pilihan;
```

7. Delete Acara (Delete First/Delete After/Delete Last)



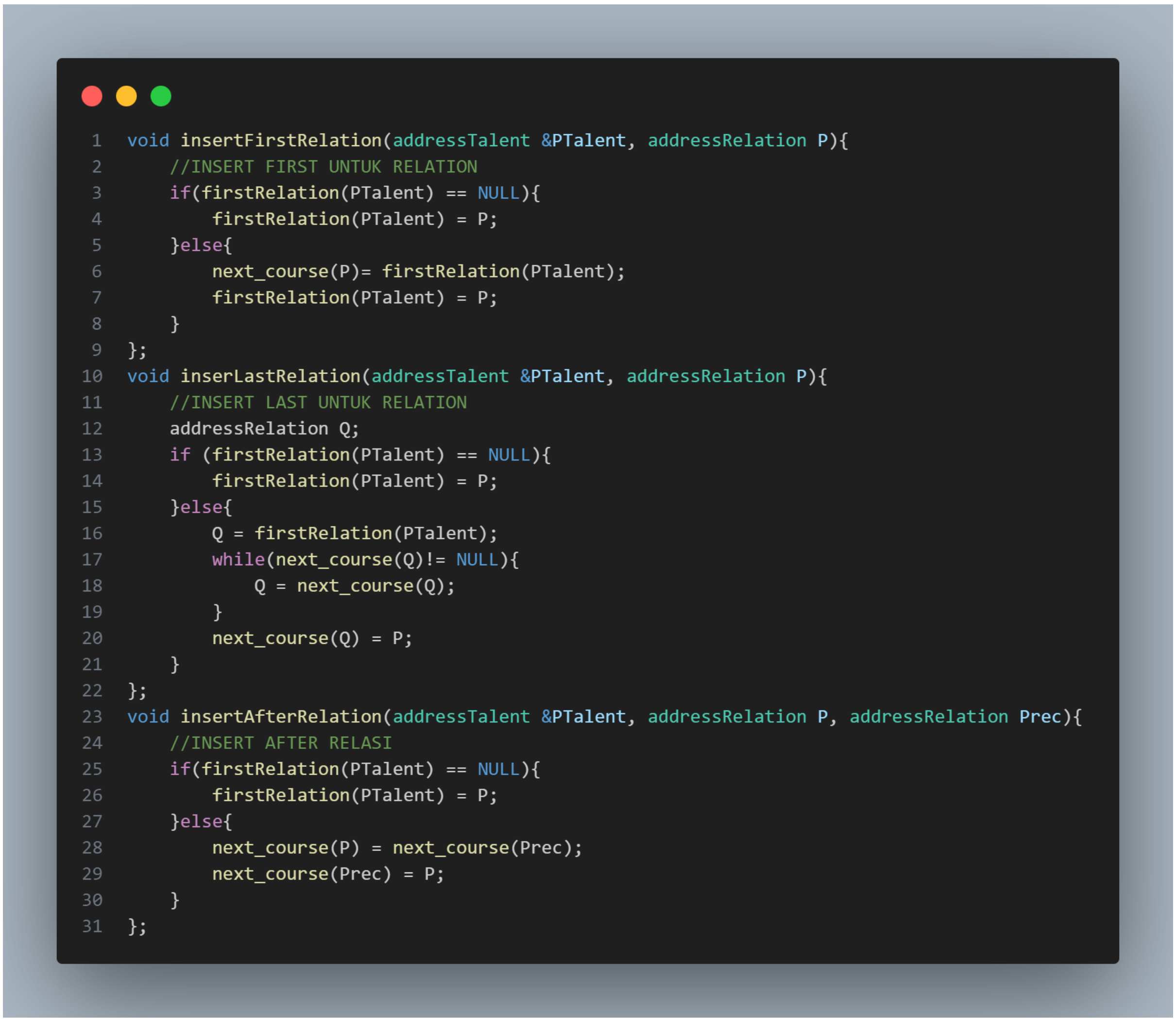
```
void deleteFirstAcara(ListAcara &LAcara, addressAcara &P, ListTalent &LTalent){
    //DELETE FIRST UNTUK LIST ACARA
    addressTalent QTalent;
    addressRelation QRelation, DRelation;
    if (first(LAcara)== NULL){
        cout<<"List ACARA KOSONG"<<endl;
    }else{
        P = first(LAcara);
        QTalent = first(LTalent);
                 while(QTalent != NULL){
            if(firstRelation(QTalent) != NULL){
                QRelation= firstRelation(QTalent);
                while (QRelation != NULL){
                    if(next_acara(QRelation) == P ){
                        if(QRelation == firstRelation(QTalent)){
                            deleteFirstRelation(QTalent);
                        }else if(next_course(QRelation) == NULL){
                            deleteLastRelation(QTalent);
                        }else {
                            DRelation = firstRelation(QTalent);
                            while(next_course(DRelation) != QRelation){
                                DRelation = next_course(DRelation);
                            }
                            deleteAfteRelation(QTalent,DRelation);
                        }
                    }
                QRelation = next_course(QRelation);
                }
            }
            QTalent = next(QTalent);
        }
        first(LAcara) = next(P);
        next(P) = NULL;
    }
};
void deleteLastAcara(ListAcara &LAcara, addressAcara &P, ListTalent &LTalent){
    //DEELTE LAST UNTUK LIST ACARA
    addressTalent QTalent;
    addressRelation QRelation, DRelation;
    addressAcara Q;
    Q = first(LAcara);
    if(first(LAcara)== NULL){
        cout<<"List ACARA KOSONG"<<endl;
    }else{
        while(next(next(Q)) != NULL){
            Q = next(Q);
        }
        P = next(Q);
        QTalent = first(LTalent);
         while(QTalent != NULL){
            if(firstRelation(QTalent) != NULL){
                QRelation= firstRelation(QTalent);
                while (QRelation != NULL){
                    if(next_acara(QRelation) == P ){
                        if(QRelation == firstRelation(QTalent)){
                            deleteFirstRelation(QTalent);
                        }else if(next_course(QRelation) == NULL){
                            deleteLastRelation(QTalent);
                        }else {
                            DRelation = firstRelation(QTalent);
                            while(next_course(DRelation) != QRelation){
                                DRelation = next_course(DRelation);
                            }
                            deleteAfteRelation(QTalent,DRelation);
                        }
                    }
                QRelation = next_course(QRelation);
                }
            }
            QTalent = next(QTalent);
        }
        next(Q) = NULL;
    }
};
void deleteAfterAcara(ListAcara &LAcara, addressAcara &P, addressAcara Prec, ListTalent &LTalent){
    //DELETE AFTER UNTUK LIST ACARA
    addressTalent QTalent;
    addressRelation QRelation, DRelation;
    if(first(LAcara) == NULL){
        cout<<"List ACARA KOSONG"<<endl;
    }else{
        P = next(Prec);
        QTalent = first(LTalent);
          while(QTalent != NULL){
            if(firstRelation(QTalent) != NULL){
                QRelation= firstRelation(QTalent);
                while (QRelation != NULL){
                    if(next_acara(QRelation) == P ){
                        if(QRelation == firstRelation(QTalent)){
                            deleteFirstRelation(QTalent);
                        }else if(next_course(QRelation) == NULL){
                            deleteLastRelation(QTalent);
                        }else {
                            DRelation = firstRelation(QTalent);
                            while(next_course(DRelation) != QRelation){
                                DRelation = next_course(DRelation);
                            }
                            deleteAfteRelation(QTalent,DRelation);
                        }
                    }
                QRelation = next_course(QRelation);
                }
            }
            QTalent = next(QTalent);
        }
        next(Prec) = next(P);
        next(P) = NULL;
    }
};
```

8. Show All Acara



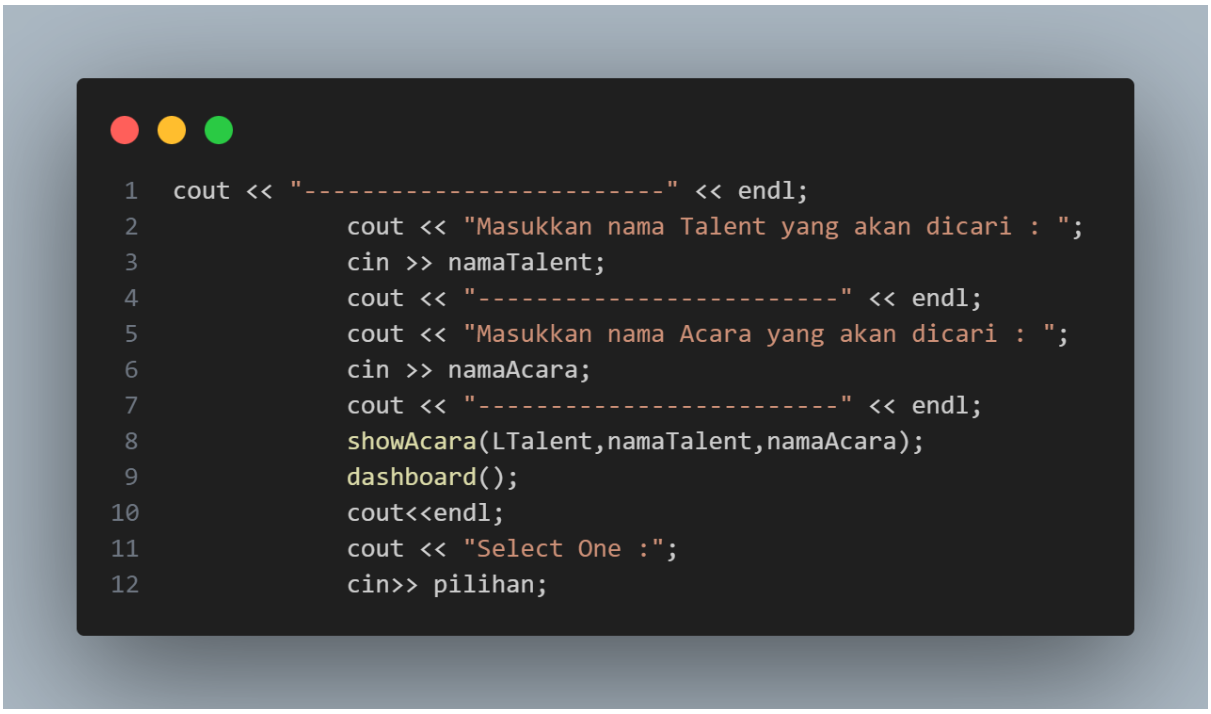
```
void printAcara(ListAcara LAcara){
    //MENGELUARKAN SEMUA INFO UNTUK LIST ACARA
    addressAcara Q;
    Q = first(LAcara);
    if(first(LAcara) == NULL){
        cout<<"LIST ACARA KOSONG"<<endl;
    }else{
        while(Q != NULL){
            cout<<"- Nama Acara :"<<info(Q).namaAcara<<endl;
            cout<<"  Tanggal    :"<<info(Q).tanggal<<endl;
            cout<<"  Tempat     :"<<info(Q).tempat<<endl;
            cout<<endl;
            Q = next(Q);
        }
    }
};
```

9. Hubungkan Relasi Talent dan Acara (Insert First/Insert After/Insert Last)

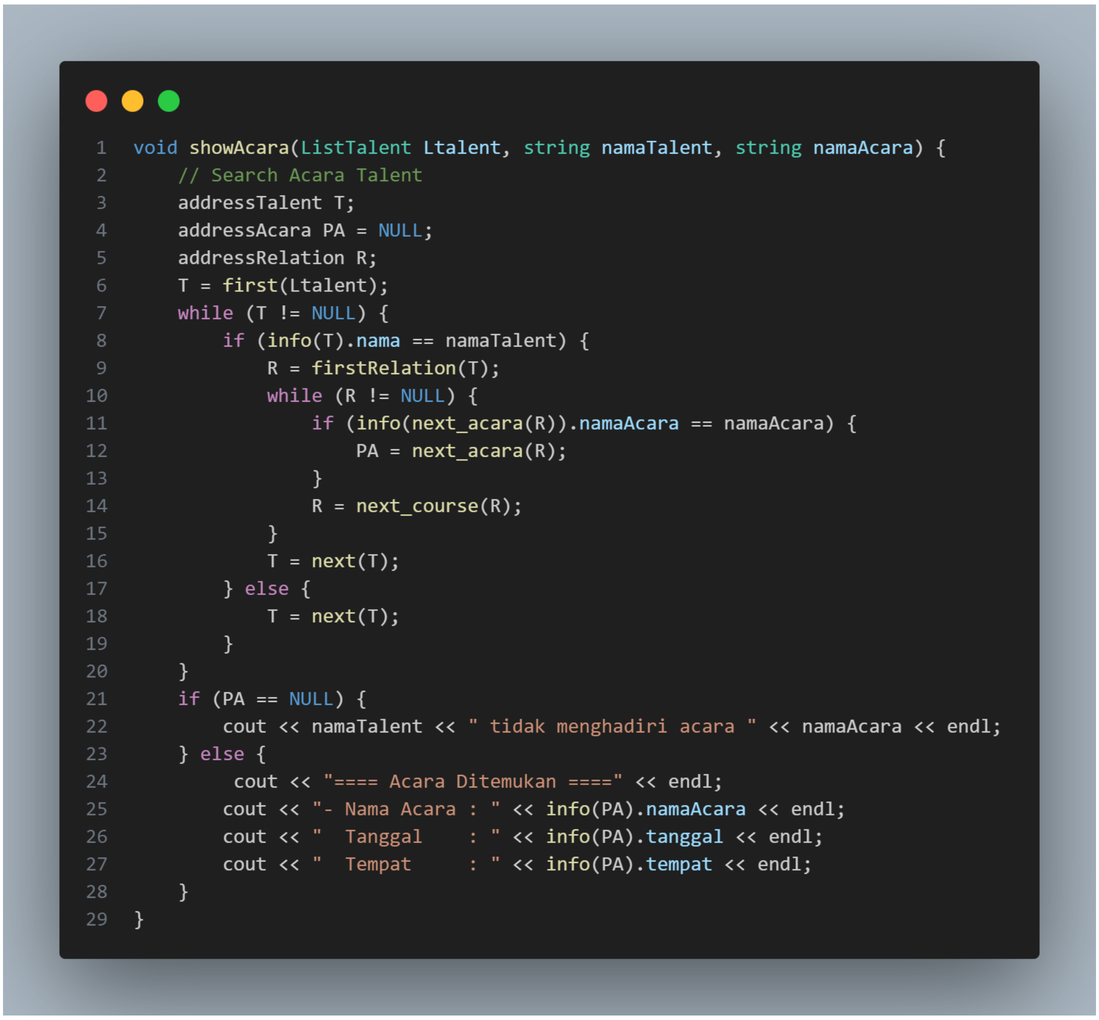


```
void insertFirstRelation(addressTalent &PTalent, addressRelation P){
    //INSERT FIRST UNTUK RELATION
    if(firstRelation(PTalent) == NULL){
        firstRelation(PTalent) = P;
    }else{
        next_course(P)= firstRelation(PTalent);
        firstRelation(PTalent) = P;
    }
};
void inserLastRelation(addressTalent &PTalent, addressRelation P){
    //INSERT LAST UNTUK RELATION
    addressRelation Q;
    if (firstRelation(PTalent) == NULL){
        firstRelation(PTalent) = P;
    }else{
        Q = firstRelation(PTalent);
        while(next_course(Q)!= NULL){
            Q = next_course(Q);
        }
        next_course(Q) = P;
    }
};
void insertAfterRelation(addressTalent &PTalent, addressRelation P, addressRelation Prec){
    //INSERT AFTER RELASI
    if(firstRelation(PTalent) == NULL){
        firstRelation(PTalent) = P;
    }else{
        next_course(P) = next_course(Prec);
        next_course(Prec) = P;
    }
};
```

10. Search Acara pada Talent

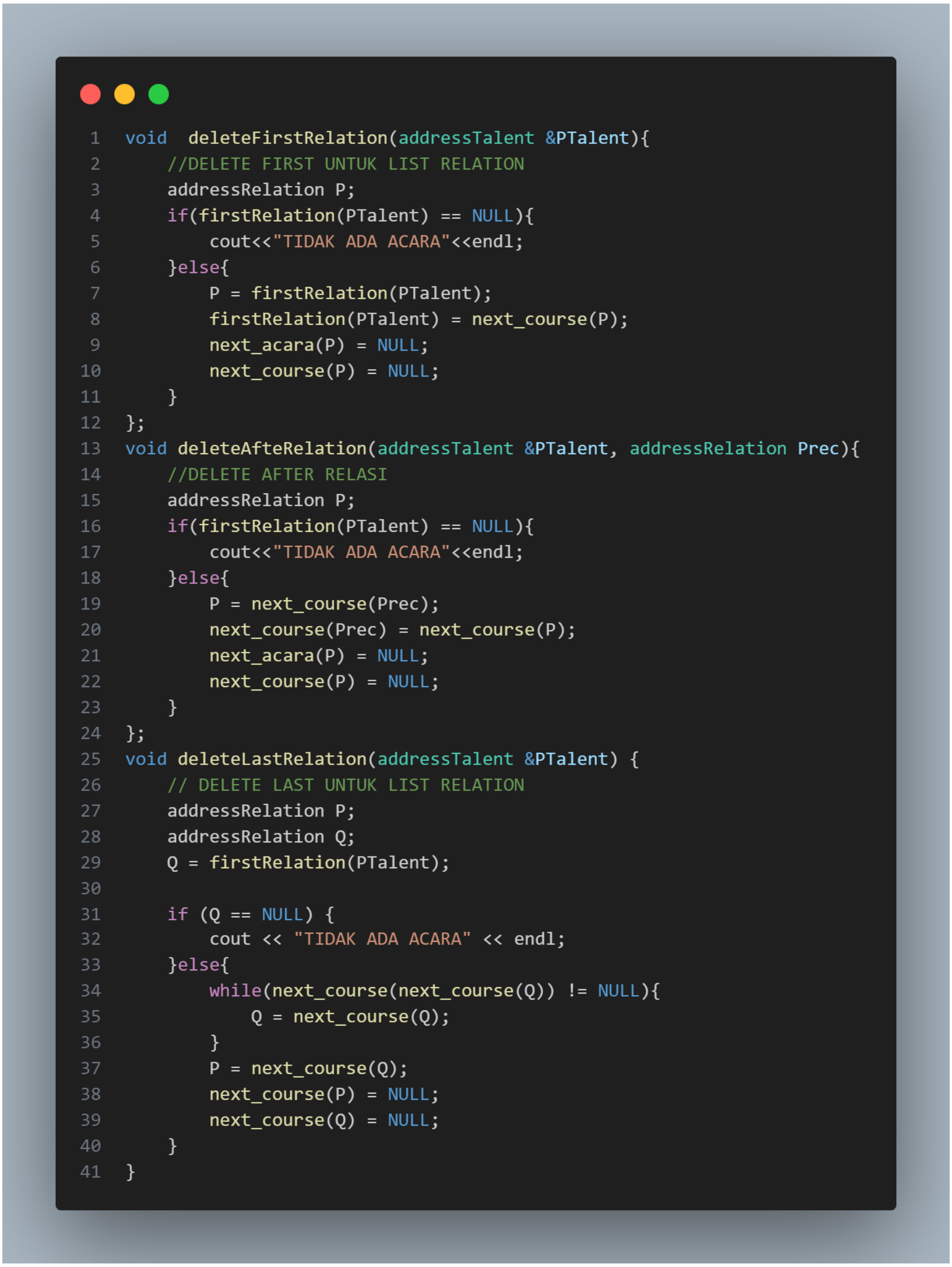


```
cout << "-------------------------" << endl;
            cout << "Masukkan nama Talent yang akan dicari : ";
            cin >> namaTalent;
            cout << "-------------------------" << endl;
            cout << "Masukkan nama Acara yang akan dicari : ";
            cin >> namaAcara;
            cout << "-------------------------" << endl;
            showAcara(LTalent,namaTalent,namaAcara);
            dashboard();
            cout<<endl;
            cout << "Select One :";
            cin>> pilihan;
```



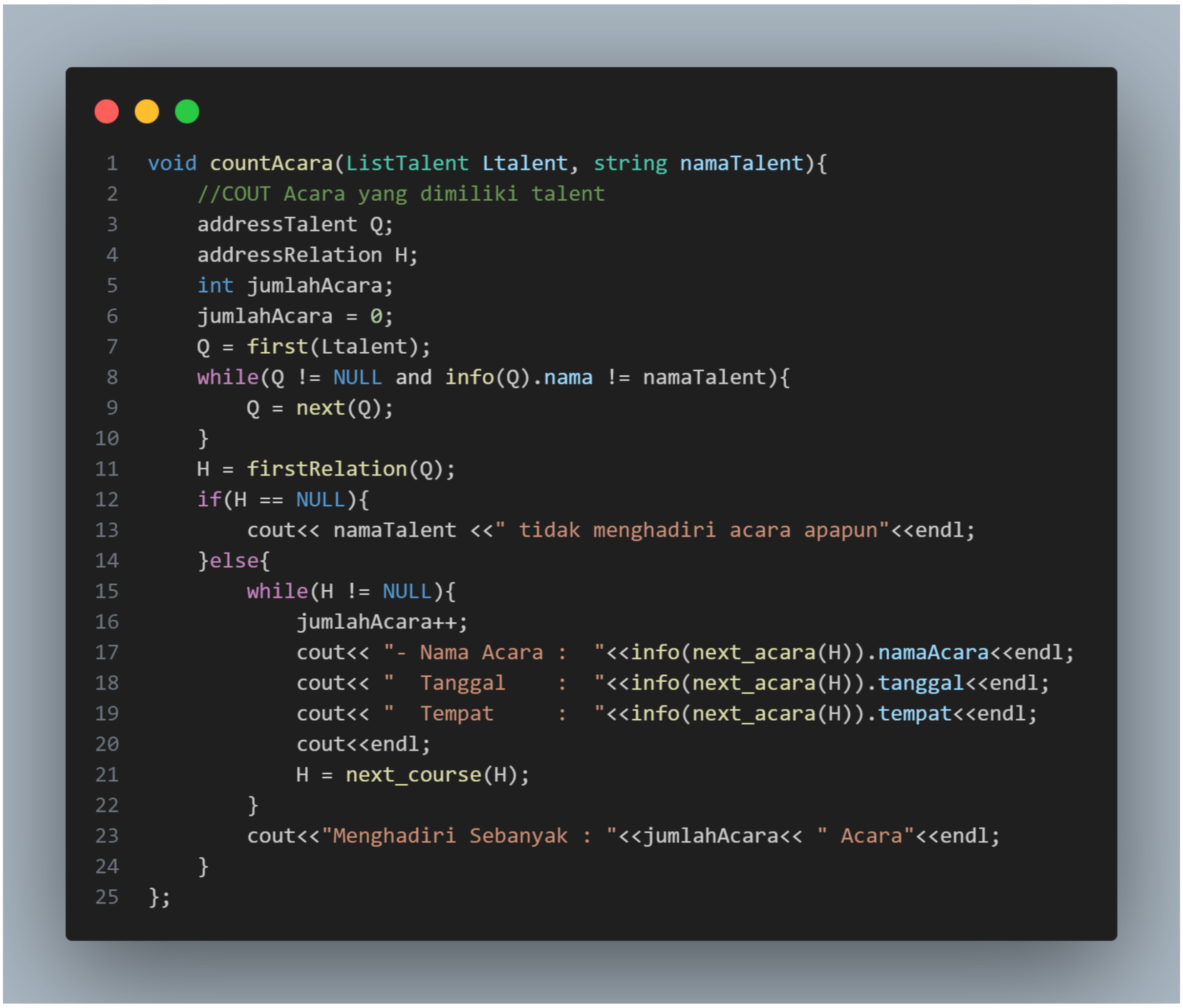
```
void showAcara(ListTalent Ltalent, string namaTalent, string namaAcara) {
    // Search Acara Talent
    addressTalent T;
    addressAcara PA = NULL;
    addressRelation R;
    T = first(Ltalent);
    while (T != NULL) {
        if (info(T).nama == namaTalent) {
            R = firstRelation(T);
            while (R != NULL) {
                if (info(next_acara(R)).namaAcara == namaAcara) {
                    PA = next_acara(R);
                }
                R = next_course(R);
            }
            T = next(T);
        } else {
            T = next(T);
        }
    }
    if (PA == NULL) {
        cout << namaTalent << " tidak menghadiri acara " << namaAcara << endl;
    } else {
         cout << "==== Acara Ditemukan ====" << endl;
        cout << "- Nama Acara : " << info(PA).namaAcara << endl;
        cout << "  Tanggal    : " << info(PA).tanggal << endl;
        cout << "  Tempat     : " << info(PA).tempat << endl;
    }
}
```

11. Delete Acara pada Talent (Delete First/Delete After/Delete Last)



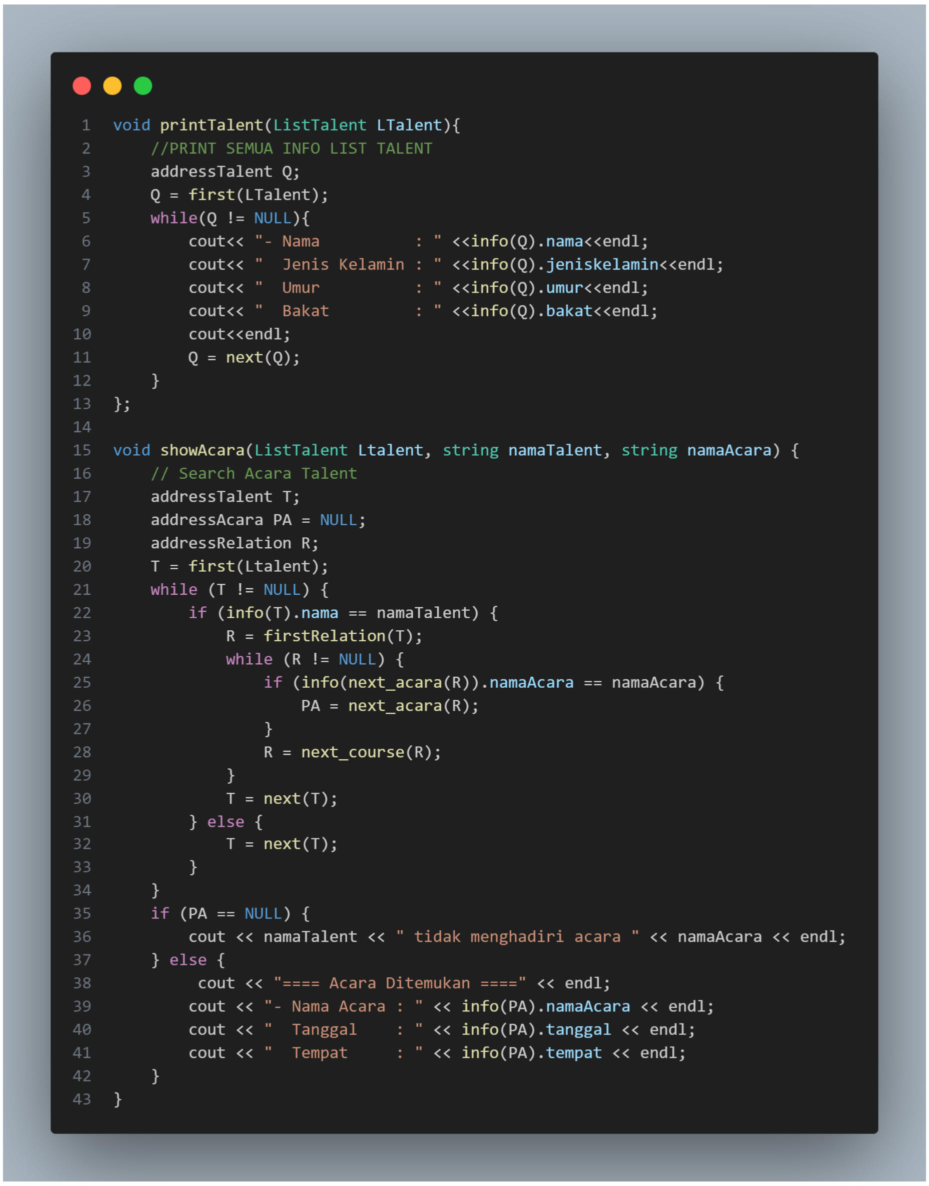
```
void  deleteFirstRelation(addressTalent &PTalent){
    //DELETE FIRST UNTUK LIST RELATION
    addressRelation P;
    if(firstRelation(PTalent) == NULL){
        cout<<"TIDAK ADA ACARA"<<endl;
    }else{
        P = firstRelation(PTalent);
        firstRelation(PTalent) = next_course(P);
        next_acara(P) = NULL;
        next_course(P) = NULL;
    }
};
void deleteAfteRelation(addressTalent &PTalent, addressRelation Prec){
    //DELETE AFTER RELASI
    addressRelation P;
    if(firstRelation(PTalent) == NULL){
        cout<<"TIDAK ADA ACARA"<<endl;
    }else{
        P = next_course(Prec);
        next_course(Prec) = next_course(P);
        next_acara(P) = NULL;
        next_course(P) = NULL;
    }
};
void deleteLastRelation(addressTalent &PTalent) {
    // DELETE LAST UNTUK LIST RELATION
    addressRelation P;
    addressRelation Q;
    Q = firstRelation(PTalent);

    if (Q == NULL) {
        cout << "TIDAK ADA ACARA" << endl;
    }else{
        while(next_course(next_course(Q)) != NULL){
            Q = next_course(Q);
        }
        P = next_course(Q);
        next_course(P) = NULL;
        next_course(Q) = NULL;
    }
}
```

12. Count Acara pada Talent

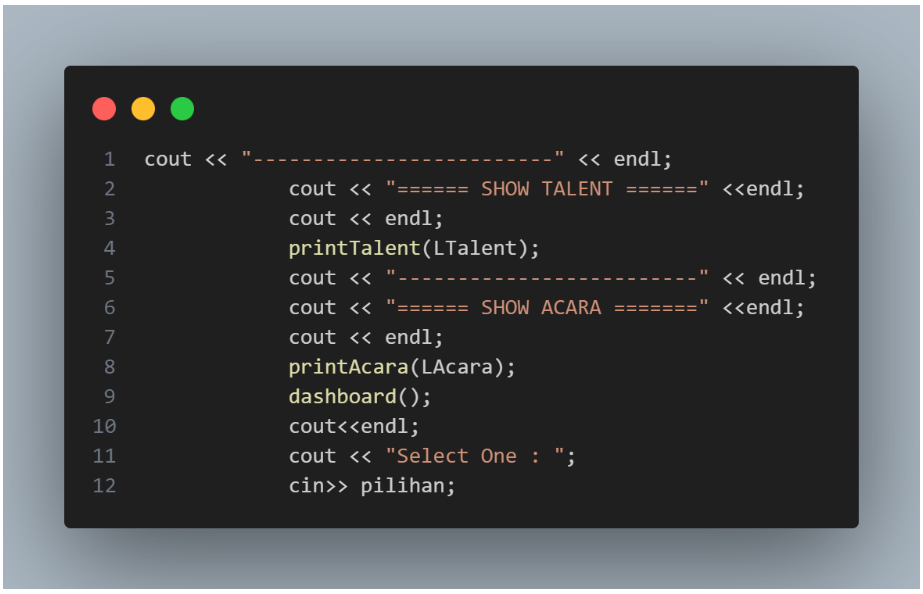


```cpp
void countAcara(ListTalent Ltalent, string namaTalent){
    //COUT Acara yang dimiliki talent
    addressTalent Q;
    addressRelation H;
    int jumlahAcara;
    jumlahAcara = 0;
    Q = first(Ltalent);
    while(Q != NULL and info(Q).nama != namaTalent){
        Q = next(Q);
    }
    H = firstRelation(Q);
    if(H == NULL){
        cout<< namaTalent <<" tidak menghadiri acara apapun"<<endl;
    }else{
        while(H != NULL){
            jumlahAcara++;
            cout<< "- Nama Acara :  "<<info(next_acara(H)).namaAcara<<endl;
            cout<< "  Tanggal    :  "<<info(next_acara(H)).tanggal<<endl;
            cout<< "  Tempat     :  "<<info(next_acara(H)).tempat<<endl;
            cout<<endl;
            H = next_course(H);
        }
        cout<<"Menghadiri Sebanyak : "<<jumlahAcara<< " Acara"<<endl;
    }
};
```

13. Show semua Talent dan acara yang ada

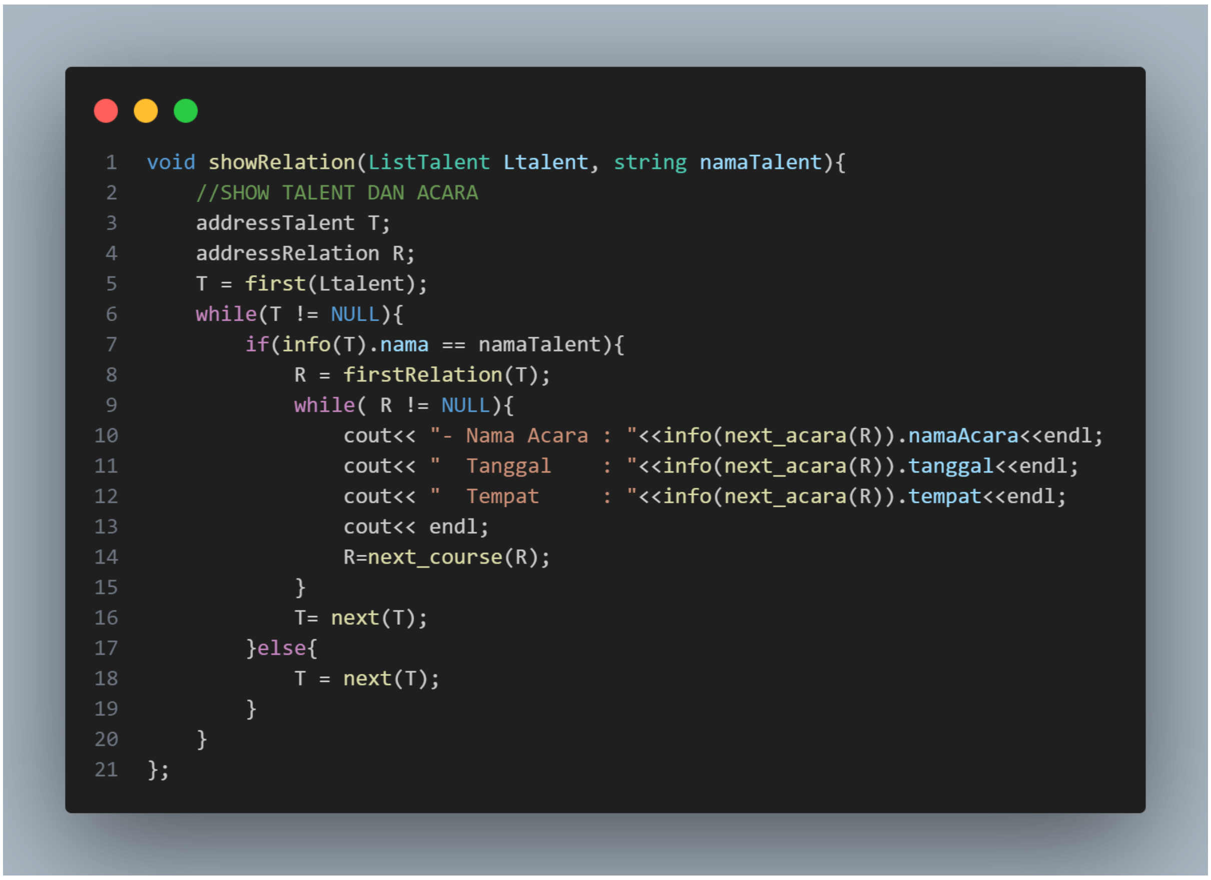


```cpp
void printTalent(ListTalent LTalent){
    //PRINT SEMUA INFO LIST TALENT
    addressTalent Q;
    Q = first(LTalent);
    while(Q != NULL){
        cout<< "- Nama          : " <<info(Q).nama<<endl;
        cout<< "  Jenis Kelamin : " <<info(Q).jeniskelamin<<endl;
        cout<< "  Umur          : " <<info(Q).umur<<endl;
        cout<< "  Bakat         : " <<info(Q).bakat<<endl;
        cout<<endl;
        Q = next(Q);
    }
};

void showAcara(ListTalent Ltalent, string namaTalent, string namaAcara) {
    // Search Acara Talent
    addressTalent T;
    addressAcara PA = NULL;
    addressRelation R;
    T = first(Ltalent);
    while (T != NULL) {
        if (info(T).nama == namaTalent) {
            R = firstRelation(T);
            while (R != NULL) {
                if (info(next_acara(R)).namaAcara == namaAcara) {
                    PA = next_acara(R);
                }
                R = next_course(R);
            }
            T = next(T);
        } else {
            T = next(T);
        }
    }
    if (PA == NULL) {
        cout << namaTalent << " tidak menghadiri acara " << namaAcara << endl;
    } else {
         cout << "==== Acara Ditemukan ====" << endl;
        cout << "- Nama Acara : " << info(PA).namaAcara << endl;
        cout << "  Tanggal    : " << info(PA).tanggal << endl;
        cout << "  Tempat     : " << info(PA).tempat << endl;
    }
}
```



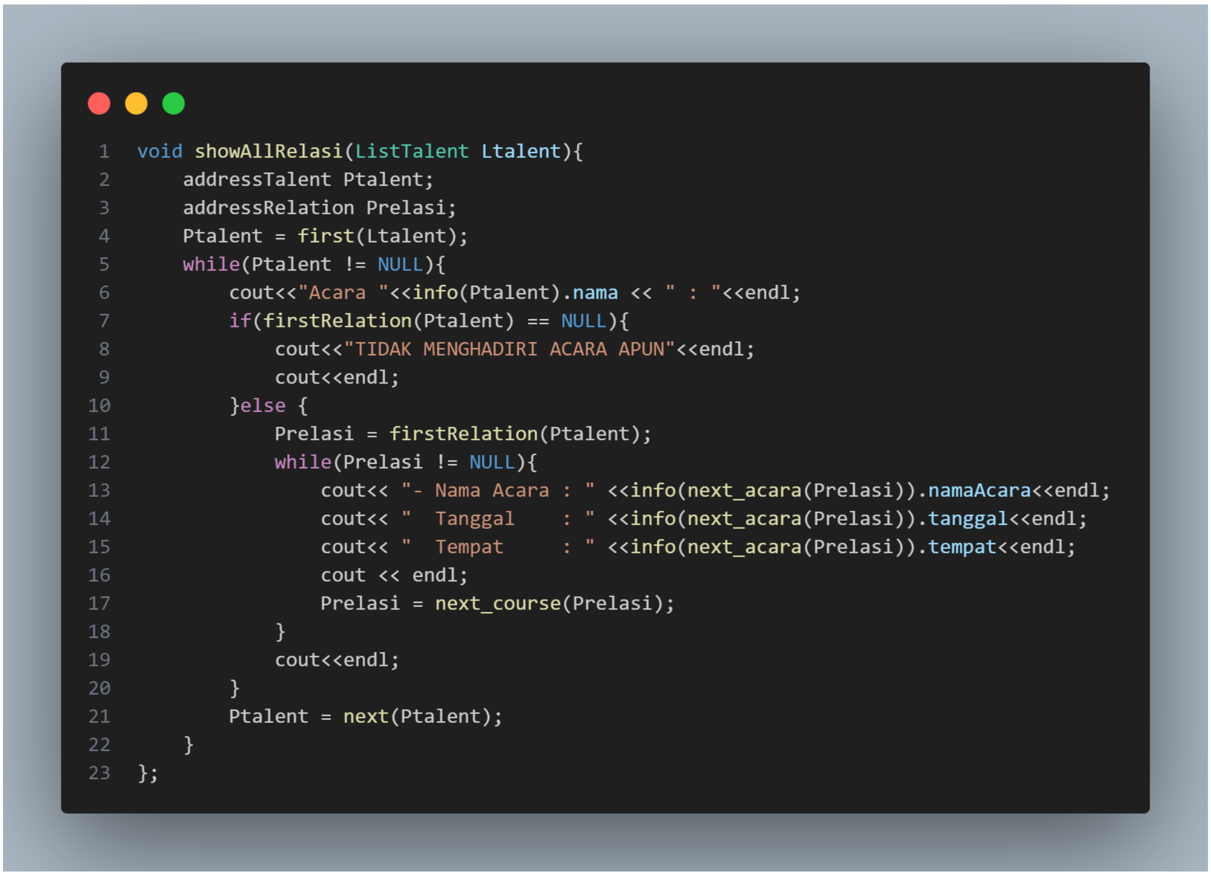
```cpp
cout << "--------------------------" << endl;
            cout << "====== SHOW TALENT ======" <<endl;
            cout << endl;
            printTalent(LTalent);
            cout << "--------------------------" << endl;
            cout << "====== SHOW ACARA =======" <<endl;
            cout << endl;
            printAcara(LAcara);
            dashboard();
            cout<<endl;
            cout << "Select One : ";
            cin>> pilihan;
```

14. Show Acara yang dimiliki Talent



```
void showRelation(ListTalent Ltalent, string namaTalent){
    //SHOW TALENT DAN ACARA
    addressTalent T;
    addressRelation R;
    T = first(Ltalent);
    while(T != NULL){
        if(info(T).nama == namaTalent){
            R = firstRelation(T);
            while( R != NULL){
                cout<< "- Nama Acara : "<<info(next_acara(R)).namaAcara<<endl;
                cout<< "  Tanggal    : "<<info(next_acara(R)).tanggal<<endl;
                cout<< "  Tempat     : "<<info(next_acara(R)).tempat<<endl;
                cout<< endl;
                R=next_course(R);
            }
            T= next(T);
        }else{
            T = next(T);
        }
    }
};
```

15. Show Semua Relasi



```
void showAllRelasi(ListTalent Ltalent){
    addressTalent Ptalent;
    addressRelation Prelasi;
    Ptalent = first(Ltalent);
    while(Ptalent != NULL){
        cout<<"Acara "<<info(Ptalent).nama << " : "<<endl;
        if(firstRelation(Ptalent) == NULL){
            cout<<"TIDAK MENGHADIRI ACARA APUN"<<endl;
            cout<<endl;
        }else {
            Prelasi = firstRelation(Ptalent);
            while(Prelasi != NULL){
                cout<< "- Nama Acara : " <<info(next_acara(Prelasi)).namaAcara<<endl;
                cout<< "  Tanggal    : " <<info(next_acara(Prelasi)).tanggal<<endl;
                cout<< "  Tempat     : " <<info(next_acara(Prelasi)).tempat<<endl;
                cout << endl;
                Prelasi = next_course(Prelasi);
            }
            cout<<endl;
        }
        Ptalent = next(Ptalent);
    }
};
```