

```

#include <iostream>
#include <cstring>

using namespace std;

struct treno{
    char destinazione[31];
    int ore;
    int minuti;
    treno *pun;
};

struct TabTreni{
    char stazione[31];
    int quanti;
    treno *testa;
};

void inizializzaTabTreni(TabTreni &T, const char* src){
    if(strlen(src)>30)
        return;
    strcpy(T.stazione, src);
    T.testa = NULL;
    T.quanti = 0;
}

bool nuovoTreno(TabTreni &T, int h, int m, const char* dest) {
    if (h < 0 || h > 23 || m < 0 || m > 59)
        return false;

    if (strlen(dest) > 30)
        return false;

    treno *p, *q;
    for (p = T.testa; p != NULL; p = p->pun) {
        if (p->ore == h && p->minuti == m)
            return false;
    }

    treno *r = new treno;
    r->ore = h;
    r->minuti = m;
    strcpy(r->destinazione, dest);

    for (p = T.testa; p != NULL && p->ore < h; p = p->pun) {
        q = p;
    }

    if (p != NULL && p->ore == h) {
        for (; p != NULL && p->ore == h && p->minuti < m; p =
p->pun) {
            q = p;
        }
    }

    if (p == T.testa) {

```

```

        r->pun = T.testa;
        T.testa = r;
    } else {
        r->pun = p;
        q->pun = r;
    }
    T.quantiti++;
    return true;
}

bool cancellaTreno(TabTreni &T, int h, int m, const char* dest)
{
    if (h < 0 || h > 23 || m < 0 || m > 59)
        return false;

    if (strlen(dest) > 30)
        return false;

    treno *p, *q;
    for (p = T.testa; p != NULL; p = p->pun) {
        if (p->ore == h && p->minuti == m && strcmp(p->
destinazione, dest) == 0)
            break;
        q = p;
    }

    if (p==NULL)
        return false;

    if (p==T.testa) {
        T.testa = T.testa->pun;
        delete p;
    }
    else {
        q->pun = p->pun;
        delete p;
    }

    T.quantiti--;
    return true;
}

bool partenzaTreno(TabTreni &T) {
    if (T.testa == NULL)
        return false;

    treno *p = T.testa;
    T.testa = T.testa->pun;
    delete p;

    T.quantiti--;
    return true;
}

int quantiTreni(TabTreni &T) {
    return T.quantiti;
}

```

```

void stampaTabTreni(TabTreni &T) {
    cout << "Treni in partenza da " << T.stazione << endl;
    if(T.testa != NULL) {
        for(treno *p = T.testa; p!=NULL; p=p->pun) {
            cout << "[" << p->ore << ":" << p->minuti << "];"
            cout << p->destinazione << endl;
        }
    }
}

bool domanda_due(int *v, int *w, int dim) {
    for (int i = 0; i < dim; i++) {
        bool trovato = false;
        for (int j = 0; j < dim; j++) {
            if (v[i] == w[j]) {
                trovato = true;
            }
        }
        if (!trovato)
            return false;
    }
    return true;
}

int domanda_tre(int n){
    if(n==0)
        return 0;

    return n*n+domanda_tre(n-1);
}

int main() {
    TabTreni T;

    inizializzaTabTreni(T, "PISA");
    nuovoTreno(T, 10, 30, "TORINO");
    nuovoTreno(T, 10, 45, "MILANO");
    nuovoTreno(T, 9, 45, "TORINO");
    nuovoTreno(T, 9, 30, "ROMA");
    nuovoTreno(T, 10, 45, "ROMA"); // FALLISCE

    stampaTabTreni(T);

    cancellaTreno(T, 10, 30, "TORINO");

    stampaTabTreni(T);

    partenzaTreno(T);

    stampaTabTreni(T);

    int m = quantiTreni(T);
    cout << "m: " << m << endl;

    int v[] = {3, 5, 4, 2};
}

```

```

    int w[] = {2, 3, 5, 4};

    bool b = domanda_due(v, w, 4);
    cout << b << endl;

    int n = domanda_tre(3);
    cout << n << endl;

    return 0;
}

// domanda1 : conversione -12 su 8 bit: 11110100

```