Problema: data una stringa, contare le occorenze di ogni singolo carattere.

Per risolvere il problema utilizzo un vettore di contatori, uno per ogni
possibile carattere del codice ascii.

I contatori sono inizializzati a zero, direttamente nella dichiarazione,
nel seguente modo:

```
int c[256] = {0};
```

La funzione che risolve il problema è la seguente:

```
        void contaCaratteri( char *s, int c[] ) {
                int i;

                for(i=0; s[i]!='\0'; ++i)
                        c[ s[i] ]++;
        }
```

Supponiamo ora di avere la seguente struttura dati:

```
#define DS 512

#define DE 5000

typedef char studente[DS];
typedef studente elenco[DE];
```

elenco è terminato da una stringa "vuota", cioè una stringa di lunghezza 0

Vogliamo contare quanti studenti sono nati in ogni mese dell'anno:

```
typedef int contatori[13];



        void contaStudentiMese( elenco e, contatori c ){
                int i;
                char d[8];

                for(i=0; strlen(e[i])>0; ++i)
                        c[ mese( data(e[i], d) ) ]++;
        }


        char * data(studente s, char d[]){
                int i,k c=3;
```

```
        for(i=0; c>0; ++i)
                if(s[i]==':') c--;

        for(i=i+1, k=0; k<8; k++, i++)
                d[k]=s[i];

        return d;
}

int mese(char d[]) {
        return converti(&d[2], 2);
}
```

Nuovo problema: dato un vettore di numeri, non necessariamente distinti, determinare le posizioni del minimo.

La funzione che risolve il problema ha in ingresso il vettore dei numeri,
la dimensione, ed un vettore nel quale inserire le posizioni.
In uscita restituirà la dimensione del vettore delle posizioni.

```
int posizioneMinimo(int v[], int dim, int p[]){
        int i, k=0, min;

        min=minimo(v, dim);

        for(i=0; i<dim; ++i)
                if(v[i] == min) p[k++]=i;

        return k;
}
```

Un problema analogo è determinare i più vecchi tra gli studenti dell'elenco.

```
int posizionePiuVecchi(elenco e, int p[]){
        int i;
        char dmin[8];

        minimo(e, dmin);

        for(i=0; strlen(e[i])>0; ++i)
                if( uguale(e[i], dmin) ) p[k++]=i;

        return k;
}


char *minimo(elenco e, char d[]){
        int i;
        char tmp[8];
```

```
                data(e[0], d);
                for(i=1; strlen(e[i])>0; ++i)
                        if( dataCmp(data(e[i], tmp), d) < 0 )
                                data(e[i], d);
                return d;
        }
```

Nuovo problema: ordinare un vettore di interi.

```
        void ordina(int v[], int dim) {
                int i, k, p;

                for(i=0; i<dim; ++i) {
                        p=i;
                        for(k=i+1; k<dim; k++)
                                if(v[k]<v[p]) p=k;
                        scambio(&v[i],&v[p]);
                }
        }
```

Utilizzando lo stesso algoritmo, ordinare elenco in base alla data di nascita.

```
void ordinaElencoPerEta(elenco e){
        int i, k, p;
        char tmp1[8], tmp2[8];

        for(i=0; strlen(e[i])>0; ++i) {
                p=i;
                for(k=i+1; strlen(e[k])>0; ++k)
                        if(dataCmp(data(e[i], tmp1),data(e[k],
tmp2))<0)
                                p=k;
                scambio(e[i], e[k]);
        }
}
```