Classe 4 B i
26-01-2002

Siano:

```
Data dataCrea( int, int, int );
Data dataCreaNulla( void );

int giorno( Data );
int mese( Data );
int anno( Data );

int isDataNulla( Data );
void dataPrint( Data );
int dataCmp( Data, Data );
```

le primitive che operano su Data.

A. Risolvere i seguenti problemi, producendo una breve descrizione del problema, l'algoritmo e la codifica in c:

   1. Data una data d determinare la data di d + n.
   2. Dato un vettore di date di nascita determinare la data del più vecchio.
   3. Dato un vettore di date contare quante sono quelle di un determinato anno.
   4. Dato un vettore di date di nascita determinare quanti sono i maggiorenni.
   5. Copiare in un vettore nuovo tutte le date che hanno un determinato anno.

B. Sia alunno un vettore di stringhe contenente gli alunni di una determinata scuola e nascita le corrispondenti date di nascita.
   Risolvere i seguenti problemi producendo una breve descrizione del problema, l'algoritmo e la codifica in c:

   1. Dato un alunno stampare la relativa data di nascita.
   2. Ordinare gli elenchi in base alla data di nascita.
   3. Data una data restituire l'elenco degli alunni nati in quella determinata data
   4. Stampare in ordine alfabetico l'elenco degli alunni maggiorenni

Valutazione: punti 1.5 per esercizio.

A1. Realizziamo la funzione piuUno che avuto in ingresso una data restituisce la data successiva.

```
Data piuUno(Data d) {
        char tab_giorni[2][13] = {
              {0, 31,28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
              {0, 31,29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
        };
        int gg = giorno(d), mm = mese(d), aa = anno(d);

        gg++;
        if(gg > giorni[bisestile(aa), mm] ) {
                gg=1; mm++;
                if(mm > 12) { mm = 1; aa++; }
        }
        return creaData(gg, mm, aa);
}
```

Il problema si risolve richiamando per n volte la funzione piuUno.

```
Data piuN(Data d, int n) {
        int i;
        Data x = creaData( giorno(d), mese(d), anno(d) );

        for(i=0; i<n; i++)
                x = piuUno(x);
        return x;
}
```

A2. E' un problema di massimo/minimo. L'algoritmo è conosciuto. Si percorre tutto il vettore, e quando si trova una data precedente a quella del minimo la si sostituisce. Il risultato è la data del più vecchio. Il vettore è terminato da una dataNulla.

```
Data vecchio( Data d[] ) {
        Data old = dataCrea(giorno(d[0]), mese(d[0]), anno(d[0]) );
        int i;

        for(i=0; ! isDataNulla(d[i]); i++)
                if(dataCmp(old, d[i]) > 0) old = dataCrea(giorno(d[0]), mese(d[0]), anno(d[0]) );
        return old;
}
```

A3. Si percorre tutto il vettore e quando si trova una data con anno uguale all'anno in ingresso la si conteggia. Il vettore è terminato da una dataNulla

```
int conta( Data d[], int aa ) {
        int c = 0, i;

        for(i=0; ! isDataNulla(d[i]); i++)
                if( anno(d[i]) == aa ) c++;

        return c;
}
```

A4. Supponiamo di avere a disposizione una funzione che permetta di conoscere la data odierna. Chiamiamo questa funzione gettata. Possiamo allora sottrarre 18 agli anni ed avere la data dalla quale si è maggiorenni. Allora si percorre tutto il vettore e si conteggiano tutte le date più vecchie di quella trovata. Il vettore è terminato da una dataNulla

```
Int maggiorenni( Data d[] ) {
        Data x = creaData( giorn(getData()), mese(gettata()), anno(gettata()) -18);

        for(i=0; ! isDataNulla(d[i]); i++)
                if( dataCmp(d[i], x) <= 0 ) c++;
        return c;
}
```

A5. Si percorre tutto il vettore e si copia in un nuovo quando si trova una data con anno uguale a quello fornito in ingresso. Il nuovo vettore viene terminato da dataNulla. In output si ha il numero di elementi del nuovo vettore. Il vettore è terminato da una dataNulla

```
int copia( Data d[], Data nuovo[], int aa ) {
        int k, i;

        for(i=0, k=0; ! isDataNulla(d[i]); i++)
                if( anno(d[i]) == aa ) nuovo[k++] = creaData(giorno(d[i]), mese(d[i]), anno(d[i]);

        return k;
}
```

B. Il vettore di stringhe contiene in ogni cella il cognome ed il nome dell'alunno, separati da un carattere definito. Supponiamo di avere a disposizione due funzioni, *cognome* e *nome* che restituiscono una stringa contenente rispettivamente il cognome ed il nome. La data della persona di posizione i dell'elenco si trova alla posizione i delle date. L'elenco è terminato da una stringa nulla, la data da una data nulla.

B1. E' un problema di ricerca.

```
char* cognome(char *c, char s);      / * copia il cognome da s a c e restituisce c  */
char* nome(char *n, char *s);        /* copia il nome da s a n e restituisce n  */
```

cono contiene il cognome e il nome da ricercare, separati da :.

```
Data trova(char ele[][256], Data* d, char* cono) {
        int i;

        for(i=0; strlen( ele[i] ) > 0; i++)
                if(nomeCmp(ele[i], cono) = = 0) return d[i];
        return creaDataNulla();
}

int nomeCmp(char *s1, char *s2) {
        char c1[256], c2[256], n1[256], n2[256];
        int uguale;

        if( (uguale=strcmp(cognome(c1, s1), cognome(c2, s2)) ) = = 0)
                return strcmp(nome(n1, s1), nome(n2, s2));
        return uguale;
}
```

B2. E' un problema di ordinamento. Quando si scambiano i nomi, si devono anche scambiare le date.

```
void ordina( char ele[][256], Data *d ) {
        int i, k, pmin;

        for(i=0;strlen(ele[i]) > 0; i++) {
                pmin = i;
                for(k=k+1; strlen(ele[k]) > 0; k++)
                        if( dataCmp( d[k], d[pmin] ) < 0 ) pmin=k;
                swapNome(ele[i], ele[pmin] );
                swapData(&d[i], &d[pmin] );
        }

        void swapNome(char* x, char*y) {
                char z[256] ;

                strcpy(z, x) ;
                strcpy(x, y) ;
                strcpy(y, z) ;
        }
```

```
void swapData(Data *d1, Data *d2) {
        Data z = creaData(giorno(d1), mese(d1), anno(d1));

        *d1= creaData(giorno( *d2 ), mese( *d2 ), anno( *d2 ));
        *d2= creaData(giorno(z), mese(z), anno(z));
}
```

B3. E' necessario copiare in un nuovo elenco tutti gli alunni nati in una determinata data. Si ricerca tra le date ed ogni volta che coincidono le date, si copia il cognome e il nome nel nuovo elenco.

```
int copia( char ele[][256], char nuovo[][256], Data *d, Data x) {
        int i, k;

        for(i=0, k=0; ! dataNulla(d[i]); i++)
                if(dataCmp(d[i], x) = = 0) strcpy(nuovo[k++], ele[i]);
        nuovo[k][0]='\0';
        return k;
}
```

B4. E' necessario copiare in un nuovo elenco gli alunni maggiorenni. L'inserimento nel nuovo elenco viene effettuato rispettando l'ordinamento.

```
int copia( char ele[][256], char nuovo[][256], Data *d,) {
        int i, k;
        nuovo[0][0]='\0';

        for(i=0, k=0; ! dataNulla(d[i]); i++)
                if(maggiorenne(d[i]) insert(nuovo, ele[i], k++);
        return k;
}

void insert(char x[][256], char *s, int i) {

        for( ; i>0 && nomeCmp(x[i-1], s) >0; i--)
                strcpy(x[i], x[i-1]);
        strcpy(x[i], s);
}
```