Programmazione I (Tucci/Distasi)

PR1 MT/RD 15/02/2022

Modello: 1

Cognome: _____

Nome: _____

Matricola: _____

Email: _____

Regole del gioco: Compilare i dati personali prima d'incominciare. Alla fine della prova, inviare un singolo file pdf (non immagini separate, non un link web) all'indirizzo email

`prog1unisa.aula1@gmail.com`

Buon lavoro!

1. Consideriamo un file contenente i voti riportati dagli studenti di una classe. Ogni studente può avere un numero variabile di voti. I voti sono numeri interi compresi fra 1 e 10, separati da spazi; la sequenza termina con il valore sentinella -1. Sappiamo che la classe contiene al massimo `MAXCLASSE` studenti, ma non conosciamo il numero di voti di ogni studente.

   Ecco un esempio di come potrebbe essere il file.

   ```
   maria 6  5 7 6 -1
   jessica -1
   giovanni 3 4  6 6 5 -1
   claudia   7 -1
   marco 7 9  8 -1
   ```

   Si definisca opportunamente un tipo `Studente` adatto a contenere il nome e la media dei voti. Si scriva poi una funzione

   ```
   Studente * leggi_studente(FILE *fp)
   ```

   che legga dal file già aperto i dati relativi a un solo studente, restituendo il puntatore ad un oggetto di tipo `Studente` allocato dinamicamente, contenente il nome dello studente e la sua media (oppure -1 se lo studente non ha voti). In caso di fine file, la funzione restituisce un puntatore nullo.

2. Nelle stesse ipotesi dell'esercizio precedente, si scriva una funzione

```
Studente ** leggi_classe(FILE *fp)
```

che alloca dinamicamente un array di `MAXCLASSE` puntatori a `Studente` e usa ripetutamente `leggi_studente()` per riempirlo con i dati degli studenti che hanno almeno un voto. Le posizioni usate devono essere tutte contigue. Le posizioni non usate devono essere tutte alla fine e contenere `NULL`. La funzione `leggi_classe()` restituisce l'array così riempito.

Risposte per il modello 1

1. Consideriamo un file contenente i voti riportati dagli studenti di una classe. Ogni studente può avere un numero variabile di voti. I voti sono numeri interi compresi fra 1 e 10, separati da spazi; la sequenza termina con il valore sentinella -1. Sappiamo che la classe contiene al massimo `MAXCLASSE` studenti, ma non conosciamo il numero di voti di ogni studente.

   Ecco un esempio di come potrebbe essere il file.

   ```
   maria 6  5 7 6 -1
   jessica -1
   giovanni 3 4  6 6 5 -1
   claudia   7 -1
   marco 7 9  8 -1
   ```

   Si definisca opportunamente un tipo `Studente` adatto a contenere il nome e la media dei voti. Si scriva poi una funzione

   ```
   Studente * leggi_studente(FILE *fp)
   ```

   che legga dal file già aperto i dati relativi a un solo studente, restituendo il puntatore ad un oggetto di tipo `Studente` allocato dinamicamente, contenente il nome dello studente e la sua media (oppure -1 se lo studente non ha voti). In caso di fine file, la funzione restituisce un puntatore nullo.

   Risposta
   Ecco una possibile soluzione.

```
#define MAXCLASSE 20            // max num. studenti in classe
#define MAXNAME 50              // max lunghezza di un nome

typedef struct
{
  char nome[MAXNAME];
  double media;
} Studente;

void *xmalloc(size_t nbytes)
{
  void *result = malloc(nbytes);
  if (result == NULL)
    {
      fprintf(stderr, "xmalloc(%lu) failed. Exiting.\n", nbytes);
      exit(-1);
    }
  return result;
}
```

```
Studente *leggi_studente(FILE * fp)
{
  Studente *new;                          // puntatore a studente conterra' risultato
  char nome[MAXNAME];
  int i, scanf_status, nvoti, somma_voti, voto_attuale;
  double media;

  scanf_status = fscanf(fp, "%s", nome); // leggiamo il nome
  if (scanf_status != 1)                  // fine file?
    {
      return NULL;
    }
  nvoti = somma_voti = 0;
  while (1)                               // loop: leggiamo, contiamo e sommiamo i voti
    {
      fscanf(fp, "%d", &voto_attuale);   // leggiamo
      if (voto_attuale == -1)            // voto sentinella: finito studente
        {
          break;
        }
      nvoti++;                           // contiamo
      somma_voti += voto_attuale;        // sommiamo
    }
  // finita lettura, assembliamo struct e restituiamola
  new = xmalloc(sizeof(Studente));
  if (somma_voti == 0)                    // nessun voto? media diventa -1
    {
      media = -1.0;
  } else                                  // caso normale: ci sono voti, calcoliamo media
    {
      media = (double) somma_voti / nvoti;
    }
  strcpy(new->nome, nome);
  new->media = media;
  return new;
}
```

2. Nelle stesse ipotesi dell'esercizio precedente, si scriva una funzione

   ```
   Studente ** leggi_classe(FILE *fp)
   ```

   che alloca dinamicamente un array di `MAXCLASSE` puntatori a `Studente` e usa ripetutamente `leggi_studente()` per riempirlo con i dati degli studenti che hanno almeno un voto. Le posizioni usate devono essere tutte contigue. Le posizioni non usate devono essere tutte alla fine e contenere `NULL`. La funzione `leggi_classe()` restituisce l'array così riempito.

   Risposta
   Ecco una possibile soluzione.

```
Studente **leggi_classe(FILE * fp)
{
  Studente **classe;                    // sara' array di puntatori
  Studente *pstudente;                  // puntatore a studente appena letto
  int i;

  classe = xmalloc(sizeof(Studente *) * MAXCLASSE); // spazio per MAXCLASSE puntatori
  i = 0;
  while (i < MAXCLASSE)
    {
      pstudente = leggi_studente(fp);
      if (pstudente != NULL)            // file non finito, lettura OK?
        {
          if (pstudente->media != -1)   // saltiamo studenti senza voto
            {
              classe[i++] = pstudente;  // nuovo studente va in classe[]
            }
      } else                            // file finito?
        {
          break;
        }
    }
  while (i < MAXCLASSE)                 // riempiamo di NULL fino in fondo
    {
      classe[i++] = NULL;
    }
  return classe;
}
```

Programmazione I (Tucci/Distasi)

PR1 MT/RD 15/02/2022

Modello: 2

Cognome: _____

Nome: _____

Matricola: _____

Email: _____

Regole del gioco: Compilare i dati personali prima d'incominciare. Alla fine della prova, inviare un singolo file pdf (non immagini separate, non un link web) all'indirizzo email

`prog1unisa.aula2@gmail.com`

Buon lavoro!

1. Consideriamo un file contenente i voti riportati dagli studenti di una classe. Ogni studente può avere un numero variabile di voti. I voti sono numeri interi compresi fra 1 e 10, separati da spazi; la sequenza termina con il valore sentinella -1. Sappiamo che la classe contiene al massimo `MAXCLASSE` studenti, ma non conosciamo il numero di voti di ogni studente.

   Ecco un esempio di come potrebbe essere il file.

   ```
   maria 6  5 7 6 -1
   jessica -1
   giovanni 3 4  6 6 5 -1
   claudia   7 -1
   marco 7 9  8 -1
   ```

   Si definisca opportunamente un tipo `Studente` adatto a contenere il nome e la media dei voti. Si scriva poi una funzione

   ```
   Studente * leggi_studente(FILE *fp)
   ```

   che legga dal file già aperto i dati relativi a un solo studente, restituendo il puntatore ad un oggetto di tipo `Studente` allocato dinamicamente, contenente il nome dello studente e la sua media (oppure -1 se lo studente non ha voti). In caso di fine file, la funzione restituisce un puntatore nullo.

2. Nelle stesse ipotesi dell'esercizio precedente, si scriva una funzione

```
Studente ** leggi_classe(FILE *fp)
```

che alloca dinamicamente un array di `MAXCLASSE` puntatori a `Studente` e usa ripetutamente `leggi_studente()` per riempirlo con i dati degli studenti che hanno almeno un voto. Le posizioni usate devono essere tutte contigue. Le posizioni non usate devono essere tutte alla fine e contenere `NULL`. La funzione `leggi_classe()` restituisce l'array così riempito.

Risposte per il modello 2

1. Consideriamo un file contenente i voti riportati dagli studenti di una classe. Ogni studente può avere un numero variabile di voti. I voti sono numeri interi compresi fra 1 e 10, separati da spazi; la sequenza termina con il valore sentinella -1. Sappiamo che la classe contiene al massimo `MAXCLASSE` studenti, ma non conosciamo il numero di voti di ogni studente.

   Ecco un esempio di come potrebbe essere il file.

   ```
   maria 6  5 7 6 -1
   jessica -1
   giovanni 3 4  6 6 5 -1
   claudia   7 -1
   marco 7 9  8 -1
   ```

   Si definisca opportunamente un tipo `Studente` adatto a contenere il nome e la media dei voti. Si scriva poi una funzione

   ```
   Studente * leggi_studente(FILE *fp)
   ```

   che legga dal file già aperto i dati relativi a un solo studente, restituendo il puntatore ad un oggetto di tipo `Studente` allocato dinamicamente, contenente il nome dello studente e la sua media (oppure -1 se lo studente non ha voti). In caso di fine file, la funzione restituisce un puntatore nullo.

   Risposta
   Ecco una possibile soluzione.

```
#define MAXCLASSE 20            // max num. studenti in classe
#define MAXNAME 50              // max lunghezza di un nome

typedef struct
{
  char nome[MAXNAME];
  double media;
} Studente;

void *xmalloc(size_t nbytes)
{
  void *result = malloc(nbytes);
  if (result == NULL)
    {
      fprintf(stderr, "xmalloc(%lu) failed. Exiting.\n", nbytes);
      exit(-1);
    }
  return result;
}
```

```
Studente *leggi_studente(FILE * fp)
{
  Studente *new;                          // puntatore a studente conterra' risultato
  char nome[MAXNAME];
  int i, scanf_status, nvoti, somma_voti, voto_attuale;
  double media;

  scanf_status = fscanf(fp, "%s", nome); // leggiamo il nome
  if (scanf_status != 1)                  // fine file?
    {
      return NULL;
    }
  nvoti = somma_voti = 0;
  while (1)                               // loop: leggiamo, contiamo e sommiamo i voti
    {
      fscanf(fp, "%d", &voto_attuale);   // leggiamo
      if (voto_attuale == -1)            // voto sentinella: finito studente
        {
          break;
        }
      nvoti++;                           // contiamo
      somma_voti += voto_attuale;        // sommiamo
    }
  // finita lettura, assembliamo struct e restituiamola
  new = xmalloc(sizeof(Studente));
  if (somma_voti == 0)                    // nessun voto? media diventa -1
    {
      media = -1.0;
  } else                                  // caso normale: ci sono voti, calcoliamo media
    {
      media = (double) somma_voti / nvoti;
    }
  strcpy(new->nome, nome);
  new->media = media;
  return new;
}
```

2. Nelle stesse ipotesi dell'esercizio precedente, si scriva una funzione

   ```
   Studente ** leggi_classe(FILE *fp)
   ```

   che alloca dinamicamente un array di `MAXCLASSE` puntatori a `Studente` e usa ripetutamente `leggi_studente()` per riempirlo con i dati degli studenti che hanno almeno un voto. Le posizioni usate devono essere tutte contigue. Le posizioni non usate devono essere tutte alla fine e contenere `NULL`. La funzione `leggi_classe()` restituisce l'array così riempito.

   Risposta
   Ecco una possibile soluzione.

```
Studente **leggi_classe(FILE * fp)
{
  Studente **classe;                    // sara' array di puntatori
  Studente *pstudente;                  // puntatore a studente appena letto
  int i;

  classe = xmalloc(sizeof(Studente *) * MAXCLASSE); // spazio per MAXCLASSE puntatori
  i = 0;
  while (i < MAXCLASSE)
    {
      pstudente = leggi_studente(fp);
      if (pstudente != NULL)            // file non finito, lettura OK?
        {
          if (pstudente->media != -1)   // saltiamo studenti senza voto
            {
              classe[i++] = pstudente;  // nuovo studente va in classe[]
            }
      } else                            // file finito?
        {
          break;
        }
    }
  while (i < MAXCLASSE)                 // riempiamo di NULL fino in fondo
    {
      classe[i++] = NULL;
    }
  return classe;
}
```