prima prova scritta

8.11.2000

Correzione

cultura generale

- Chi può essere considerato il primo "programmatore"?
- Charles Babbage, che realizza la prima macchina "general purpose", in grado di essere programmata, nel 1824.

cultura generale

- Qual è considerato il primo calcolatore elettronico?
- ABC, costruito da Atanasoff nel 1939. Subito dopo vengono costruiti ENIAC (il primo di grandi dimensioni) e COLOSSUS (in Inghilterra).

cultura generale

- Cosa si intende per "linguaggio di alto livello"?
- Un linguaggio i cui costrutti si avvicinano al linguaggio naturale e indipendenti dalla piattaforma. Sarà compito dell'interprete o del compilatore far corrispondere ad ogni comando la corrispondente sequenza di comandi in linguaggio macchina.

cultura generale

- Perché è necessario usare compilatori diversi quando si vuole compilare ed eseguire un programma su piattaforme diverse?
- Perché un compilatore produce codice in linguaggio macchina, e quest'ultimo varia al variare delle piattaforme

tipi, espressioni, valori

- Se una variabile di tipo T ha dimensione 16 bit, cosa si può dire del tipo T?
- Si può solo dire che il numero di valori distinti appartenenti al tipo T è 2^{16}.

tipi, espressioni, valori

- Cosa si intende per "portata" o "campo di validità" di un identificatore?
- E' la porzione di codice all'interno del quale quell'identificatore è visibile ed utilizzabile

tipi, espressioni, valori

- E' vero o falso che 45 è un identificatore?
- Falso: un identificatore non può avere come primo carattere una cifra.

tipi, espressioni, valori

- Che differenza c'è tra una costante definita con **#define** ed una variabile definita con l'attributo **const**?
- **#define** indica al preprocessore di attuare una sostituzione sintattica, che modifica il file sorgente: non avviene nessuna allocazione di memoria (come invece avviene nel caso di una variabile **const**).
- Una variabile **const** può essere locale.

tipi, espressioni, valori

- Cosa si intende per ambiente?
- E' una struttura dinamica che rappresenta ad ogni passo della computazione la corrispondenza tra identificatori (nomi) e variabili (locazioni di memoria)

comandi

- Cosa si intende per comando semplice?
- Un'espressione seguita dal carattere ';'.

- Fare un esempio di comando non semplice
- if (a<0) a++;
- while (x!= 'z') printf("%c",x);
- eccetera

espressioni aritmetiche

- Applicando le regole di precedenza tra gli operatori aritmetici e le regole di casting implicito si calcoli il valore delle seguenti espressioni

6 + 5/4 - 3	**4**
6.0 + 5/4 -3	**4.0**
6 + 5.0/4 - 3	**4.25**

espressioni aritmetiche

- Quali sono i valori ed i tipi delle seguenti espressioni?

19 % 5	4	int
19 / 5	3	int
19 / 5.0	3.8	double

espressioni aritmetiche

- Convertire la costante binaria 10000001 in ottale
- 201

- Convertire la costante binaria 1000011 in esadecimale
- C3

espressioni

- Quali sono gli operatori relazionali definiti in C?
- `==` `>` `<` `>=` `<=` `!=` con la seguente semantica:

```
x == y    /* il valore di x è uguale a quello di y */
x > y     /* il valore di x è strettamente maggiore di quello di y */
x < y     /* il valore di x è strettamente minore di quello di y */
x >= y    /* il valore di x è maggiore o uguale a quello di y */
x <= y    /* il valore di x è minore o uguale a quello di y */
x != y    /* il valore di x è diverso da quello di y */
```

comandi

- Qual è il valore delle variabili x e y dopo l'esecuzione del comando seguente?

```
int x,y;
x=y=0;
if (x = y++)    //FALSE, x=0, y=1
  y *= ++x;
else
  y *= --x;     // y = y * (-1)
```

- x = -1 y = -1

comandi

- Qual è il valore delle variabili x e y dopo l'esecuzione del comando seguente?

```
int x,y;
x=y=0;
if (x = ++y)    // TRUE x=1, y=1
    y *= ++x;   // y = y * 2;
else
    y *= --x;
```

- x = 2, y = 2

comandi

- Il costrutto seguente è un comando in C? Perché?

```
23;
```

- Si, perché "23" è un'espressione ed un comando semplice è un'espressione seguita dal carattere ';'.

espressioni booleane

- Scrivere un'espressione booleana che testa se il valore di una variabile intera x è compreso tra 0 e 9 (estremi inclusi)
- **`x>=0 && x<=9`**

espressioni booleane

- Per quali valori di x e di y l'espressione booleana !((x==4) && (y=2*x)) è falsa?
- Tale espressione è falsa se è vera l'espressione (x==4) && (y=2*x).
  Devono essere veri i due congiunti.
  L'espressione (x==4) è vera solo per x=4.
  L'espressione (y=2*x) è vera se x!=0.
  Quindi la risposta è: **x=4 e y qualsiasi**.

espressioni booleane

- Per quali valori di x e di y l'espressione booleana ((x!=4) || (x!=18)) è vera?
- Per tutti i valori di x: x non potrà mai avere contemporaneamente il valore 4 ed il valore 18!

procedure

- Scrivere un programma che legge un numero intero N e scrive i primi N numeri dispari.
- ```
  #include <stdio.h>
  int main(void){
       int i,N;
       scanf("%d",&N);
       for (i=0; i<N; i++)
             printf("%d ", 2*i + 1);
       printf("\n");
      return (0);
  }
  ```
  ```

procedure

- Ci può essere più di un comando **return** all'interno del corpo di una procedura?
- Si.
  In particolare, se la procedura è una funzione, ci dovrà essere un comando **return** in ogni ramo del flusso di controllo.

procedure

- Una funzione può restituire più di un valore? Perché?
- No.
  Perché il tipo di ritorno di una funzione è uno solo.

procedure

- Quando viene chiamata una procedura come viene esteso l'ambiente?
- Vengono allocate tante variabili quanti sono i parametri formali; ognuna di esse ha come identificatore il corrispondente parametro formale ed è inizializzata al valore del corrispondente parametro attuale.
  Poi l'ambiente viene esteso con le variabili locali dichiarate nella procedura.

procedure

- Scrivere un programma che stampa gli interi positivi minori di 100 divisibili sia per 6 che per 7
- ```
  #include <stdio.h>
  int main(void){
      int i;
      for (i=0; i<=100; i++)
            if ( !(i%6) && !(i%7) )
                  printf("%d \n",i);
      return (0);
  }
  ```
  ```

procedure

- Scrivere una funzione ricorsiva che, dato un intero N, scrive a video N linee vuote

```
int scrivi(int n){
    if (n>0){
          printf("\n");
          scrivi(n-1);
    }
    return (n);
}
```

procedure

- Scrivere una funzione ricorsiva che, dato un intero positivo K, scrive a video 2K caratteri '*'.

```
int scrivi(int k){
    if (k>0){
          printf("**");
          scrivi(k-1);
    }
    return (k);
}
```

terminazione

- Per quali valori dei parametri attuali la seguente procedura termina correttamente?
- Quali sono i possibili valori restituiti?

```
int foo(int a, int b){
    int z=0;
    if (b=0)
      return 3;
    else{
      while (a==3)
        z=foo(a,2*b);
      return z;
    }
}
```

Termina se a !=3

In tal caso

restituisce 0

terminazione

- Per quali valori dei parametri attuali la seguente procedura termina correttamente?
- Quali sono i possibili valori restituiti?

```
int foo(int a, int b){
    int z=0;
    if (b=0)
      return 3;
    else{
      while (a!=3)
        z=foo(a,2*b);
      return z;
    }
}
```

Termina se a = 3

In tal caso

restituisce 0

procedure

- Scrivere una funzione che, data una tabella **a** di interi e la sua dimensione **dim**, restituisce il numero di elementi della tabella che hanno valore strettamente minore di 0.

```
int conta_negativi(int a[], int dim){
  int i;
  int totale=0;
  for (i=0; i<dim; i++){
    if (a[i]<0)
      totale++;
  }
  return (totale);
}
```

procedure

- Scrivere una funzione che data una tabella **a** di interi e la sua dimensione **dim**, restituisce il numero di elementi della tabella che hanno valore uguale a 0.

```
int conta_zeri(int a[], int dim){
  int i;
  int totale=0;
  for (i=0; i<dim; i++){
    if (a[i]==0)
      totale++;
  }
  return (totale);
}
```

tabelle

- La seguente procedura produce un errore a tempo di esecuzione. Perché?

```
int foo(int a[], int dim){
  int z;
  int risultato = 1;
  for (z=0; z<dim; z++,a[z]=z)
    risultato = risultato * a[z];
  return risultato;
}
```

- Perché alla fine dell'ultima iterazione viene assegnato il valore **dim** ad **a[dim]**, oltrepassando i limiti della tabella.

tabelle

- La seguente procedura produce un errore a tempo di esecuzione. Perché?

```
int foo(int a[], int dim){
  int z;
  int risultato = 1;
  for (z=0; a[z]!=0 && z<dim; z++)
    risultato = risultato * a[z];
  return risultato;
}
```

- Perché quando **z=dim**, viene valutata l'espressione **a[dim]**, oltrepassando i limiti della tabella, prima di eseguire il test **z<dim**.

tabelle

- Dichiarare una variabile tabella di 4 interi, inizializzandola assegnando ai 4 elementi della tabella i valori 1,2,3,4, rispettivamente.
- **`int tabella[]={1,2,3,4};`**

- Dichiarare una variabile tabella di 5 caratteri, inizializzandola assegnando ai 5 elementi della tabella i valori 'a','b','c','d','e', rispettivamente.
- **`char tabella[]={'a','b','c','d','e'};`**