Esercitazioni di Fondamenti di Informatica - Lez. 4

24/10/2019

Principali errori di codifica

1- rientro del testo o indentazione

don't

```
int main()
{int a=0;


if(a==0)  {
a=b;
}


…

if(a>0)  {
     ...
}
     else  {
          ...
     }
```

do

```
int main()
{
     int a=0;
     …

     if(a==0)
     {
        a=b;
     }

     if(a>0)
     {
        …
     }
     else
     {
        ...
     }
}
```

do (*alternativa*)

```
int main(){
     int a=0;
     …

     if(a==0){
        a=b;
     }

     if(a>0){
        …
     }
     else{
        ...
     }
}
```

2- errori logici o di costrutto

don't

```
if(a>0)
{
     …
}
else  (a<=0)
{
     …
}
```

La negazione di **a>0** è già minore o uguale a 0.
Difatti è inutile anche:

```
if(a>0)
{
    …
}
else
{
    if (a<0)
    {
       …
    }
     else
    {
       if (a==0)
```

do

```
if(a>0)
{
     …
}
else
{
     …
}


if(a>0)
{
    …
}
else
{
    if (a<0)
    {
       …
    }
     else
    {
       …
```

3- errori logici di assegnazione

don't

```
int a=1, b=2;
…..
if (a = 0)
{
}

..
if (b = a)
{
}

..

int trovato = 0;
while(trovato = 0)
{
}
```

Quanto vale qui *b* alla fine?
ATTENZIONE: "=" assegna, "==" confronta.

do

```
int a=1,b=2;
…..
if (a == 0)
{
}

..
if (a == b)
{
}

..

int trovato = 0;
while(trovato == 0)
{
}
```

scanf("%d", A[i]); ----- ricorda & --> scanf("%d", &A[i]);

scanf("%c", c); --char ha bisogno di uno spazio -> scanf(" %c", c);

4- errori sui cicli

don't

```
while(x>0);
{
}



for (x=1; x<100; x++);
{
}
```

Il ";" crea un blocco di statement del ciclo vuoto

do

```
while(x>0)
{
}



for (x=1; x<100; x++)
{
}
```

```
while (i < n) {
  if (n % i == 0) {
    ndiv = ndiv + 1;
    i = i + 1;
   }
 }
```

```
while (i < n) {
  if (n % i == 0) {
  ndiv = ndiv + 1;
  }
 i = i + 1;
}
```

Esercizi sui tipi in C

I codici e le soluzioni degli esercizi sono nella cartella **codice4**

1. **Dato il seguente codice, trovare gli errori, correggerli e indicare cosa si ottiene come output.**

```
0 #include<stdio.h>
1
2 int main(){
3 int i= 65;
4 int j = 'A';
5 int k="$";
6
7 int u=i+k;
8 char v = 122;
9
10  printf("%d %d %d %d %c %d %c\n",i ,j ,k,u,u,v,v);
11 }
```

Soluzione

```
0 #include<stdio.h>
1
2 int main(){
3 int i = 65;
4 int j = 'A';
5 int k= '$';     <---con apici indica il valore ASCII
6
7 int u=i+k;
8 char v = 122;
9
10  printf("%d %d %d %d %c %d %c\n",i ,j ,k,u,u,v,v);
11 }
```

Il programma stampa:
_ Il numero 65
_ Il codice ASCII (intero) del carattere 'A'
_ Il codice ASCII (intero) del carattere '$'
_ Il numero dato dalla somma tra 65 e il codice ASCII (intero) del carattere '$'
_ il carattere corrispondente al codice ASCII dato dal numero precedente
_ il numero 122
_ il carattere corrispondente al codice ASCII 122

2. **Si scriva un programma che dati in ingresso N intero > 0 produce in uscita un triangolo isoscele di N righe**
   (nota: uso di cicli innestati, una per le righe ed una per le colonne)

   esempio
   Inserire la lunghezza del lato: 4

```
*
**
***
****
```

```c
#include <stdio.h>
int main() {
int n, i, j;
i = 0, j = 0;
printf("Inserire la lunghezza del lato: ");
scanf("%d", &n);
while (i < n) {
   while (j <= i) {
       printf("*");
       j++;
   }
   printf("\n");
   j = 0;
   i++;
}
return 0;
}
```

3. **Definisci un nuovo tipo di dato contenente un intero, un carattere e un numero reale, in tre modi diversi:**
   (A) definire una struttura senza definire un nuovo tipo;
   (B) definire una struttura senza definire un nuovo tipo e con una nuova variabile;
   (C) definire una struttura definendo un nuovo tipo. Provare a copiare una variabile struttura in un altra e modificarla.
   Stampare il risultato della struttura ottenuta con ogni definizione.

```c
#include < stdio.h>

int main() {

/* modi alternativi per definire una struct */

/*A definire una struttura senza definire un nuovo tipo */
 struct A {
           int intero;
           char carattere;
           float reale;
 } ;
/* se voglio dichiarare una variabile di tipo prova devo usare
 una struct prova */

 struct A prova;
     prova.intero = 1;
     prova.crattere = 'P';
     prova.reale = 2.4;

/* stampiamo il contenuto di prova:*/
     printf("\nA.\n");
     printf("\nintero: %d\n", prova.intero);
     printf("carattere: %c\n", prova.carattere);
     printf("reale: %f\n", prova.reale);

/* B definire una struttura senza definire un nuovo tipo
 e con una nuova variabile
N.B. subito dopo la definizione della struct posso
definire una serie di variabili della struttura */

struct B {
           int intero;
           char carattere;
           float reale;
} prova2;

```

```c
 prova2.intero = 1;
 prova2.crattere = 'P';
 prova2.reale = 2.4;

    printf("\nB)\n");
    printf("\nintero: %d\n", prova2.intero);
    printf("carattere: %c\n", prova2.carattere);
    printf("reale: %f\n", prova2.reale);


/*C definire una struttura definendo un nuovo tipo */

    typedef struct{
           int intero;
           char carattere;
           float reale;
    } C;
    C prova3;

           prova3.intero = 1;
           prova3.carattere = 'P';
           prova3.reale = 0 . 2 4;

    printf("\nC)\n");
    printf("\nintero: %d\n", prova3.intero);
    printf("carattere: %c\n", prova3.carattere);
    printf("reale: %f\n", prova3.reale);

/* Provare a copiare una variabile struttura in un altra ,
    modificarla e stampare il risultato.*/
C prova4;
    prova4 = prova3;
    prova4.intero = 4356;

    printf("\nC) struttura copiata e modificata\n");
    printf("\nintero: %d\n", prova4.intero);
    printf("carattere: %c\n", prova4.carattere);
    printf("reale: %f\n", prova4.reale);
}
```

4. **Definire una struttura che permetta di contenere una serie di dati di squadre calcistiche: nome squadra, codice identificativo squadra, goal fatti, goal subiti.**
   Data una sequenza di squadre, stampare i nomi delle squadre che hanno fatti piu´ goal di quanti ne abbiano subiti
   Soluzione:

```c
#include<stdio.h>
#define MAXS 20   /* Lunghezza massima dei nomi delle squadre */
#define MAXA 6   /*  Massimo numero di squadre memorizzabili  */

/* Definisco un tipo per i campi testuali
   N.B. C non ha un tipo stringa!!! */
typedef char String [MAXS]; // qui salveremo i nomi delle squadre

/* Definisco una struttura per
   memorizzare una squadra */

typedef struct {
    String nome;
    int ID;
    int goal_fatti;
    int goal_subiti;
    } Squadra;

/* Dichiarazione della sequenza di squadre
   come array di strutture */

typedef Squadra Campionato[MAXA];
```

```c

int main(){

    Campionato campionato = {
        {"Juventus",1,10,12},
        {"Milan",8,7,6},
        {"Inter",10,13,11},
        {"SPAL",2,9,10},
        {"Pizzighettone",5,8,4},
        {"Udinese",14,5,7}
    };
    for (int i=0; i<MAXA; i++){
        if(campionato[i].goal_fatti > campionato[i].goal_subiti){
            printf("%s differenza goal: %d \n",
            campionato[i].nome,
            campionato[i].goal_fatti-campionato[i].goal_subiti);
        }

    }
}
```

5. **Definire le strutture dati per memorizzare i dati anagrafici di un gruppo di persone: nome, cognome e data di nascita. La data di nascita deve essere nel formato 01-GEN-1970. Identificare chi ´e la persona piu giovane dell'anagrafica e stamparne nome e cognome.**

```
0 #include <stdio.h>
1 #define MAXS 30  /* lunghezza massima campi testuali */ _
2 #define MAXP 3   /* _ numero massimo persone in anagrafica  */
3
4 /* Enumero tutti i mesi */
5 typedef enum {
6 GEN = 1, FEB, MAR, APR, MAG, GIU, LUG, AGO, SET, OTT, NOV, DIC
7  } Mese;
8 /* Definisco un tipo per i campi testuali */
9 typedef char string[MAXS];
10
11 /* Struttura per memorizzare una data_nascita */
12 typedef struct {
13           int giorno;
14           Mese mese;
15           int anno;
16 } Data;
17
18 /* Struttura con i dati di una persone */
19 typedef struct {
20           string nome;
21           string cognome;
22           data datanascita;
23 } Persona;
24
25 /* Dichiarazione dell 'anagrafica come collezione di strutture persona */
26 typedef Persona Anagrafica[MAXP];
27
28 int main() {
29    Anagrafica anagrafica = {
30           {"Mario", "Rossi", {10, APR, 1980}},
31           {"Filippo", "Bianchi", {10, APR, 1970 }},
32           {"Maria", "Verdi", {5, APR, 1990} }
33  };
34
35  int i;
36           /* Calcolo della persona piu' giovane */
37
38           /* Inizializzo il risultato con la  prima persona di anagrafica */
39  Persona piugiovane = anagrafica[0];
40
41           /* Cicliamo attraverso tutte le persone  contenute in anagrafica */
42  for (i = 0; i < MAXP; i++) {
43                        /* confronto sull 'anno */
44    if (piugiovne.datanascita.anno != anagrafica[i].datanascita.anno) {
45           if (anagrafica[i].datanascita.anno > piugiovane.datanascita.anno) {
46                 piugiovane = anagrafica[i];
47           }
48    }
49           /* confronto sul mese */
50
51 else {
52           if (piugiovane.datanascita.mese != anagrafica[i].datanascita.mese) {
53                 if (anagrafica[i].datanascita.mese >
piugiovane.datanascita.mese) {
54                        piugiovane = anagrafica[i];
55                 }
56           }
57 }
58 /* confronto sul giorno */
```

```
59
60 else {
61            if (piugiovane.datanascita.giorno != anagrafica[i].datanascita.giorno)
{
62                  if (anagrafica[i].datanascita.giorno >
piugiovane.datanascita.giorno) {
63                        piugiovane = anagrafica[i];
64                  }
65              }
66            }
67
68 }/* fine ciclo for */
69
70 printf{("La persona piu' giovane e' %s %s\n", piugiovane.nome,
               piugiovane.cognome);
71
72 }
```

6. **Dichiara una variabile intera lunghezza ed una stringa *s* di 30 caratteri.**
   Inverti la stringa carattere per carattere con un ciclo.
   Stampa la stringa con printf("%s", ...)

```
1  #include<stdio.h>
2  #include<string.h> /* contiene funzioni specifiche per le stringhe */
3  int main() {
4
5      char s[30];
6      char temp; /* variabile temporanea */
7      int i;
8      int lunghezza;
9
10     printf('Inserire una stringa: \n');
11     scanf('%s', s);
12
13     lunghezza = strlen(s);
14
15     for (i = 0; i < lunghezza / 2; i++) {
16           temp = s[lunghezza - i - 1];
17           s[lunghezza - i - 1] = s[i];
18           s[i] = temp;
19     }
20     printf("\n%s<------ parola invertita\n", s);
21 }
```

7. **Scrivere in C un programma che trasforma un numero in base 10 in un numero in qualsiasi base (da 1 a 16). Il programma stampa la nuova rappresentazione sullo schermo.**
   I dati forniti in input saranno due : il numero da trasformare e la base per la nuova rappresentazione.

Soluzione

```
1   #include <stdio.h>
2   #define DIM 10
3   #define MAX 16
4   int main() {
5         int i = 0 , num,b;
6         int v[DIM];
7         char simb[MAX] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};
8         printf("Inserire un numero e la sua base ");
9         scanf("%d",&num);
10        scanf("%d",&b);
11
12  /* Algoritmo di conversione in base b*/
13        while (num > 0) {
14                  v[i] = num%b;
15                  num = num/b;
16                  i++;
17        }
18  /* Stampa l'array dei resti al contrario */
19        i--;
20        printf("La rappresentazione di %d in base %d e': ",num, b);
21        while (i >= 0) {
22                  printf("%c", simb[v[i]]);
23                  i--;
24        }
25   printf("\n");
26   } // main
```

Ulteriori controlli da implementare : la correttezza dei valori inseriti e che la base inserita sia un valore compreso tra 1 e 16.

8. **Si scriva un programma che legga da input due array A di 10 elementi e B di 5 elementi.**
   Il programma stampi: "CONTIENE" se A contiene la sequenza contigua dei numeri di B.
   ES: A= [3,4,66,77,88,9,33,11, 66,100] B=[77,88,9,33,11]
   output: "CONTIENE" ma non stampa nulla per B = [77,88,9,34,11]

sooluzione

```
#include <stdio.h>

#define MAX 10
//
int main() {
    int A[MAX] = { 3,4,66,77,88,9,33,11, 66,100 };
    int B[MAX / 2] = { 77,88,9,343,11 };
    // per semplicita mettiamo il calcolo vero e proprio senza lettura / scanf...
    int i, j = 0;
    // con lettura  da input
/*
    for (i = 0; i < MAX; i++) {
           scanf("%d", &A[i]);
    }
    for (i = 0; i < MAX / 2; i++) {
           scanf("%d", &B[i]);
    }
*/
    for (i = 0; i < MAX && j < MAX / 2; i++) {
           if (A[i] == B[j]) {
                  j++;
           }
           else {
                  j = 0;
           }
    }
    if (j == MAX / 2)
           printf("CONTIENE\n");
    else
           printf("non CONTIENE\n");
    return 0;
}
```