Politecnico di Milano
Informatica B, A.A. 2021/2022
Laboratorio 2

Luca Frittoli (luca.frittoli@polimi.it)
Mirko Salaris (mirko.salaris@polimi.it)

20 Ottobre 2021

1. **Posizione di un punto rispetto a una retta** Si scriva un programma che chieda all'utente di inserire i parametri dell'equazione di una retta in un piano $y = mx + q$ e le coordinate di un punto (x_0, y_0), e comunichi se il punto appartiene alla retta o si trova sopra o sotto la retta nel piano.

Soluzione

```
#include <stdio.h>
void main()
{
    // Definizione delle variabili
    float m, q, x0, y0;
    // Inserimento di m,q
    printf("Inserisci i parametri della retta m, q: ");
    scanf("%f,%f", &m, &q);
    // Inserimento di x0, y0
    printf("Inserisci le coordinate del punto x0, y0: ");
    scanf("%f,%f", &x0, &y0);
    // Confronto tra le coordinate del punto e l'equazione della retta
    if (y0 > m*x0 + q)
    {
        printf("Il punto si trova sopra la retta");
    }
    else if (y0 < m*x0 + q)
    {
        printf("Il punto si trova sotto la retta");
    }
    else
    {
        printf("Il punto appartiene alla retta");
    }
}
```

2. **Semplici ordinamenti** Si scriva un programma che chieda all'utente di inserire tre numeri e verifichi che siano in ordine crescente.
   Bonus: permetti all'utente di inserire N numeri, con N a sua scelta.

Soluzione

Versione base:

```
#include <stdio.h>
void main()
{
    // dichiarazione di tre variabili per ospitare i tre valori
    int n1, n2, n3;
    printf("Inserisci tre numeri.\n");
    printf("n1,n2,n3: ");
    scanf("%d,%d,%d", &n1, &n2, &n3);
    // verifichiamo la condizione di "ordine crescente"
    if (n1 <= n2 && n2 <= n3)
    {
        printf("I numeri %d, %d e %d sono in ordine crescente!", n1, n2, n3);
    }
    else
    {
        printf("I numeri inseriti non sono in ordine crescente :(");
    }
    printf("\n");
}
```

3. **Controllo dell'input** Scrivere un programma che acquisisce un numero intero e verifica se questo è positivo; in caso contrario il programma stampa un messaggio di errore e ripete l'acquisizione. Una volta letto un valore valido, l'algoritmo lo visualizza in decimale e in esadecimale. Realizzare il programma in due versioni: nella versione `A` si utilizzi il costrutto `do-while`, nella versione `B` si utilizzi invece il costrutto `while`.

   Suggerimento: è possibile utilizzare il carattere di conversione `%X` per stampare in esadecimale.

Soluzione

Versione `A`

```
#include <stdio.h>
void main()
{
 int v;
 do
 {
 printf("Inserire un numero intero positivo: ");
 scanf("%d", &v);
 } while (v <= 0);

 printf("Il valore inserito e' %d in decimale, %X in esadecimale", v, v);
}
```

Versione `B`

```
#include <stdio.h>
void main()
{
    int v;
    printf("Inserire un numero intero positivo: ");
    scanf("%d", &v);
    while (v <= 0)
    {
        printf("Inserire un numero intero positivo: ");
        scanf("%d", &v);
    }

    printf("Il valore inserito e' %d in decimale, %X in esadecimale", v, v);
}
```

4. **Decrementi** Si scriva un programma che, dopo aver richiesto due interi N e d all'utente, stampi i numeri N, $N-d$, $N-2*d$, $\cdots$, separati da virgole e fermandosi al più piccolo intero **maggiore** di 0.
   Nota: eseguire i calcoli e stampare il risultato due volte, una tramite `for` e una tramite `while`. Verificare che il risultato sia identico in entrambi i calcoli.
   Esempio:
   $N = 10$, $d = 3$
   Output for: `10, 7, 4, 1`
   Output while: `10, 7, 4, 1`

   $N = 35$, $d = 6$
   Output for: `35, 29, 23, 17, 11, 5`
   Output while: `35, 29, 23, 17, 11, 5`

Soluzione

```
#include <stdio.h>
void main()
{
  // dichiarazione delle due variabili necessarie
  int N, d;
  // dichiarazione della variabile ausiliaria
  int v;
  printf("Inserisci il numero 'N' e il decremento 'd'\n");
  printf("N,d: ");
  scanf("%d,%d", &N, &d);

  // esecuzione tramite for
  for (v=N; v>0; v=v-d)
  {
    printf("%d,", v);
  }
  printf("\n");

  // esecuzione tramite while
```

```
  v=N;
  while (v>0)
  {
    printf("%d,", v);
    v = v - d;
  }
  printf("\n");
}
```

5. **Ordinazioni** Si scriva un programma per supportare un'interazione come la seguente con l'utente:

```
Quante portate vuoi ordinare? 4
Cosa desideri come portata n. 1 (A 3,50€; B 7,00 €; C 4,80€; D 9,30€)? A
Cosa desideri come portata n. 2 (A 3,50€; B 7,00 €; C 4,80€; D 9,30€)? B
Cosa desideri come portata n. 3 (A 3,50€; B 7,00 €; C 4,80€; D 9,30€)? A
Cosa desideri come portata n. 4 (A 3,50€; B 7,00 €; C 4,80€; D 9,30€)? D

Il tuo ordine costa: 23,30€
```

Vincoli: si faccia uso del costrutto `switch` per la scelta dell'elemento del menù; si verifichi che la scelta dell'utente sia una tra quelle consentite (A, B, C, D).

Soluzione

```
#include <stdio.h>
void main()
{
    int numero_di_portate;
    float costo = 0;
    char tipo_portata; // tipo portata sarà A, B, C, o D
    int i;             // variabile per il ciclo
    int errore_input;  // variabile flag per controllare l'input
    printf("Quante portate vuoi ordinare?\n");
    scanf("%d", &numero_di_portate);

    // esecuzione tramite for
    for (i = 0; i < numero_di_portate; i++)
    {
        do
        {
            errore_input = 0; // resettiamo la variabile flag
            printf("Cosa desideri come portata n. %d "
                   "(A 3,50€; B 7,00 €; C 4,80€; D 9,30€)? ",
                   i + 1);
            scanf(" %c", &tipo_portata);
            switch (tipo_portata)
            {
            case 'A':
```

```
                costo += 3.5;
                break;
            case 'B':
                costo += 7.0;
                break;
            case 'C':
                costo += 4.8;
                break;
            case 'D':
                costo += 9.3;
                break;
            default:
                errore_input = 1;
                break;
            }
        } while (errore_input);
    }
    printf("il tuo ordine costa %.2f€\n", costo);
}
```

6. **Triangoli** Si scriva un programma che stampi un triangolo rettangolo di asterischi, di base decisa dall'utente.
Esempio:

```
Input: 5
Output:
    *
    * *
    *   *
    *     *
    * * * * *
```

Bonus: si modifichi il programma in modo che controlli che l'input inserito dall'utente sia un intero ≥ 2 e, nel caso non lo sia, stampi un messaggio di errore e chieda all'utente di re-inserirlo.

Soluzione

Versione bonus:

```
#include <stdio.h>
void main()
{
    int base;
    printf("Inserisci la lunghezza della base: ");
    scanf("%d", &base);
    while (base < 2)
    {
        printf("Attenzione. Inserisci un numero intero >= 2: ");
        scanf("%d", &base);
    }
```

```
        for (int riga = 1; riga <= base; riga++)
        {
            printf("* ");
            for (int colonna = 2; colonna <= riga-1; colonna++)
            {
                if (riga == base)
                {
                    printf("* ");
                }
                else
                {
                    printf("  ");
                }
            }
            if (riga > 1)
            {
                printf("*\n");
            }
            else
            {
                printf("\n");
            }
        }
    printf("\n");
    }
```

7. **Fattoriale** Si scriva un programma che, dato in ingresso un numero N, ne calcoli il fattoriale $N!$. Il fattoriale di N è così definito: $N! = N \cdot (N-1) \cdot \ldots 1$. Si ricordi che $0! = 1$ per definizione.

Soluzione

```
    #include <stdio.h>
    void main()
    {
        int N;
        // usiamo double anche se il risultato sarà concettualmente intero
        // perché solo così possiamo gestire numeri molto grandi
        double risultato;
        printf("Inserisci un numero di cui calcolare il fattoriale: ");
        scanf("%d", &N);
        while (N <= 0)
        {
            printf("Attenzione. Inserisci un numero intero non negativo: ");
            scanf("%d", &N);
        }
        // inizializzamo a 1, perché stiamo facendo delle moltiplicazioni ripetute
        // per ora è un risultato parziale, alla fine sarà il risultato finale.
        risultato = 1;
        while (N > 1)
```

```
    {
        // nota bene: N di volta in volta diminuisce.
        risultato = risultato * N;
        N--;
    }
    // "%.0f" stampa un double (o float) con 0 cifre decimali
    printf("Il risultato è: %.0f", risultato);
}
```

8. **Moltiplicazioni** Scrivere un programma che chiede all'utente due valori interi `A` e `B`. Il programma esegue la moltiplicazione di `A` e `B` per **somme ripetute** e visualizza il risultato. Nota: l'operatore di moltiplicazione `*` può essere utilizzato solo per controllare la correttezza del risultato.
Bonus: Si supporti il calcolo con numeri interi positivi e negativi.

Soluzione

Versione bonus:

```
#include <stdio.h>
void main()
{
    int a, b;
    int risultato_negativo = 0; // variabile di flag: 0=no, 1=si
    int risultato = 0;
    printf("Inserire due numeri interi a,b: ");
    scanf("%d,%d", &a, &b);
    if (a < 0)
    {
        a = -a;
        risultato_negativo = 1;
    }
    if (b < 0)
    {
        b = -b;
        risultato_negativo = !risultato_negativo;
    }

    for (risultato = 0; b > 0; b--)
    {
        risultato = risultato + a;
    }

    if (risultato_negativo)
    {
        risultato = -risultato;
    }

    printf("Il risultato della moltiplicazione e' %d\n", risultato);
}
```

9. **Numeri primi** Si scriva un programma che chieda all'utente un numero N in input e verifichi se il numero è primo. Se è primo il programma stampa "PRIMO", altrimenti chiede un altro numero (e ripete se dovesse riverificarsi la condizione).
Bonus: si modifichi il programma in modo che, se il numero non è primo, ne stampi tutti i divisori.

Soluzione

```
#include <stdio.h>
void main()
{
    int N;
    int primo = 1; // ipotizziamo sia primo. Se scopriamo che
                   // non lo è, metteremo questa variabile a 0.
    int divisore_test = 2; // variabile che andremo a incrementare ripetutamente
    do
    {
        // ripetiamo l'inizializzazione a ogni ciclo
        primo = 1;
        divisore_test = 2;
        printf("Inserisci un numero per verificare se sia primo: ");
        scanf("%d", &N);
        while (N <= 1)
        {
            printf("Attenzione. Inserisci un numero intero maggiore di 1: ");
            scanf("%d", &N);
        }
        while (primo && divisore_test < N)
        {
            if (N % divisore_test == 0)
            {
                primo = 0;
            }
            divisore_test++;
        }
        if (primo)
            printf("PRIMO");
    } while (!primo);
    printf("\n");
}
```

10. **Media iterativa** Si scriva un programma che chieda all'utente di inserire i voti degli esami (numeri interi da 18 a 30) e ne calcoli iterativamente la media (cioè, aggiornando la media ogni volta che viene inserito un voto). Il programma termina quando viene inserito un numero fuori dal range o, alternativamente, quando sono stati inseriti 10 esami.

Soluzione

```
#define N 10
#include <stdio.h>
```

```
void main()
{
    int voto;
    int n = 0; // numero di esami inseriti
    float media = 0;
    int go = 1; // diventera' 0 quando dovremo fermarci

    do
    {
        printf("Inserire un voto: ");
        scanf("%d", &voto);
        if (voto >= 18 && voto <= 30) // voto valido, aggiorniamo la media
        {
            media = (n*media + voto) / (n+1);
            printf("media voti: %.2f\n", media);
            n++;
        }
        else        // voto non valido, fermiamo il programma in anticipo
        {
            go = 0;
        }
    }while(go == 1 && n < N); // in questo caso, continuiamo
}
```