Gabriele Di giampietro **05/11/2022**

Progetto: Analizzare, commentare e correggere con eventuali soluzioni un codice di programma in linguaggio C.

Prima parte: capire cosa fa il programma senza eseguirlo, descrivendolo.

Premessa: Il codice contiene degli errori di sintassi e logici che correggeremo nella Seconda parte, non tener conto di tali errori che posso comparire nell'immagini di questa prima parte.

Analizzando il codice del programma che ci è stato dato possiamo partire dalla prima parte in alto dove troviamo le librerie che andiamo ad includere nel nostro programma. Queste librerie, dette anche direttive di processo, contengo una serie di funzioni che servono e si possono richiamare per i nostri scopi, nel nostro caso abbiamo la libreria #include <stdio.h> (standard input output) che una delle principali e più comuni.

Dopo le librerie, scendiamo e troviamo la parte dove vengono inseriti i tipi di funzioni e le variabili che andremo ad utilizzare nel nostro programma, nel nostro caso troviamo il tipo di funzione "void" seguita dal termine che utilizzeremo per inserire la nostra variabile nel codice. Il tipo di funzione "void" non restituisce nessun risultato e in genere viene utilizzata per una procedura come ad esempio un menù a scelta o come ad esempio nel nostro programma dove inserisce una moltiplicazione e una divisione.

Come possiamo vedere nell'immagine seguente, cerchiato in rosso la libreria e i sottolineato in rosso il void:

```c
#include <stdio.h>

void menu ();
void moltiplica ();
void dividi ();
void ins_string();
```

```c
void menu ()
{
      printf ("Benvenuto,
      printf ("Come posso
      printf ("A >> Moltip
```

```c
void moltiplica ()
{
      short int  a,b = 0;
      printf ("Inserisci i due numeri
      scanf ("%f", &a);
```

```c
void dividi ()
{
        int  a,b = 0;
        printf ("Inserisci i
        scanf ("%d", &a);
```

In questo caso abbiamo anche l'inserimento di una stringa di testo, evidenziato in giallo il "char" che significa che la variabile comprende l'inserimento lettere alfabetiche, il 10 tra le parentesi quadre sono il massimo dei caratteri che posso inserire (nella memoria dell'array).

```c
void ins_string ()
{
      char stringa[10];
        printf ("Inseris
```

Il corpo del programma di solito inzia con int main (), int sta per valore numerico intero, può essere inserito ogni volta che inseriamo nuove funzioni. Importante però che il corpo e le funzioni del nostro programma siano inserite all'interno delle parenti graffe che aprono e chiudono { } anche altre sotto funzioni che

andiamo ad inserire aprendo all'interno altre parentesi graffe (importante che siano tutte chiuse alla fine di ogni operazione).

Come possiamo vedere nelle immagini successive, nel nostro programma abbiamo la funzione "switch" che ha lo scopo di valutare caso per caso i valori delle nostre variabili, quando viene inserita o si raggiunge una condizione non indicata nello "switch" allora va in una condizione di default, ad esempio nel nostro caso quando inseriamo qualcosa di diverso per la nostra moltiplicazione e divisione va nel default. Il break dopo ogni caso permette di uscire dopo che il set istruzione è stata eseguita.

Nel nostro programma troviamo anche le funzioni, molto comuni, di "printf e scanf" che servono relativamente per output: mostrare il testo sullo schermo e input: inserire le variabili di riferimento, come possiamo vedere nell'immagine sottolineate in rosso (d è deciamale, f è di tipo float può inserire i numeri reali). Mentre short int, evidenziato in giallo, rappresenta in linguaggio C il tipo short o intero corto è un tipo intero che è codificato utilizzando la codifica in complemento a 2 su 16 bit (2 byte).

Cerchiato in rosso abbiamo le parentesi graffe che contengo il corpo del programma e Int main() evidenziato in giallo:

```
int main ()
{
    char scelta = {'\0'};
    menu ();
    scanf ("%d", &scelta);

    switch (scelta)
    {
        case 'A':
        moltiplica();
        break;
        case 'B':
            dividi();
            break;
        case 'C':
            ins_string();
            break;
    }
```

```
short int  a,b = 0;
printf ("Inserisci i due numeri da moltiplicare:");
scanf ("%f", &a);
scanf ("%d", &b);
```

In fine abbiamo la funzione return 0 che di solito troviamo alla fine del codice, termina l'esecuzione della funzione e restituisce un valore al chiamante. Quindi può essere inserito anche sotto la fine di funzioni all'interno del codice.

```
return 0;
```

Possiamo dedure dall'analisi del codice di programma che ci è stato dato che sia un programma per calcolare come abbiamo già detto moltiplicazioni e divisioni, oltre all'inserimento di un testo, visualizzati come menù di scelta.

Seconda Parte: Individuare nel codice casistiche non standard e individuare eventuali errori di sintassi e logici.

Eseguendo da subito il codice, anche se il compilatore non da nessun errore l'esecuzione si avvia ma non funziona. Questo dimostra come nel codice ci siano sicuramente errori logici e di esecuzione Come possiamo vedere dall'immagine nella pagina seguente l'esecuzione parte ma non funziona.

```
Benvenuto, sono un assitente digitale, posso aiutarti a sbrigare alcuni compiti
Come posso aiutarti?
A >> Moltiplicare due numeri
B >> Dividere due numeri
C >> Inserire una stringa
A

--------------------------------
Process exited after 2.664 seconds with return value 0
Premere un tasto per continuare . . .
```

Leggendo il codice ho trovato anche altri errori anche di sintassi:

dove andiamo ad effettuare la nostra scelta iniziale, utilizziamo char ma nell'inserimento della variabile viene inserito un %d che in realtà indica i numeri interi. Nella parte di codice della moltiplicazione troviamo che nelle variabili abbiamo un uguale a zero e un short int che si usa per i numeri interi corti che non ci permette di visualizzare il nostro risultato. Nella parte di codice della divisione invece del simbolo corrispondente alla funzione di divisione abbiamo una percentuale e sulla stringa del printf che ci mostra il risultato della nostra divisione abbiamo %d, variabile che non può mostrarci il nostro resto, essendo un numero intero e potrebbe invece il resto rappresentare un numero reale molto piccolo o molto grande.

Nelle figure seguenti cerchiato in rossa possiamo vedere gli errori:

```
char scelta = {'\0'};
menu ();
scanf ("%d", &scelta);
```

```
void moltiplica ()
{
      short int  a,b = 0;
      printf ("Inserisci i due numeri da moltiplicare:");
      scanf ("%f", &a);
      scanf ("%d", &b);

      short int prodotto = a * b;

      printf ("Il prodotto tra %d e %d e': %d", a,b,prodotto);
```

```
void dividi ()
{
        int  a,b = 0;
        printf ("Inserisci il numeratore:");
        scanf ("%d", &a);
      printf ("Inserisci il denumeratore:");
        scanf ("%d", &b);

        int divisione = a % b;

        printf ("La divisione tra %d e %d e': %d", a,b,divisione);
}
```

Nell'esecuzione del programma possiamo constatare anche dei "bug", ovvero nella scelta inziale se utilizziamo un altro valore numerico o una lettera minuscola il programma va in errore. Lo stesso vale anche quando nel inserire i numeri nell'operazione aritmetiche proposte, cioè appunto inserendo una lettera il programma va in crash, lo stesso vale per l'inserimento della stringa di testo se inserissimo dei numeri oltre al fatto che abbiamo un limite di 10 caratteri da inserire. Inoltre anche se rispettassimo tutte le indicazioni

date per l’inserimento come da programma a fine esecuzione il codice non ci da nessun’altra scelta se volessimo concludere o ripetere le operazioni andando in crash, anche se proviamo solamente a premere un pulsante.

Parte finale: ho inserito i commenti direttamente sulle migliori che ho apportato al codice vedere file esercizio 5-11-2022.c