PROGETTO GIORNO 10 (05-11-2022)

L'esercizio di oggi avrà come obbiettivo quello di allenare l'osservazione critica;

Dato un codice sorgente dovremmo:

- Capire cosa fa il programma senza eseguirlo
- Individuare dal codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati)
- Individuare eventuali errori di sintassi / logici
- Proporre una soluzione per ognuno di essi

Per prima cosa, quindi, andremo ad aprire il file per leggere e comprendere la funzionalità del programma

```
void menu ()
{
        printf ("Benvenuto, sono un assitente digitale, posso aiutarti a sbrigare alcuni compiti\n");
        printf ("Come posso aiutarti?\n");
        printf ("A >> Moltiplicare due numeri\nB >> Dividere due numeri\nC >> Inserire una stringa\n");

}
```

Avremmo così compreso che il codice serve per fare delle operazioni numeriche (moltiplicazione e divisione) e per inserire una stringa di testo di 10 caratteri.

Analizzando meglio andremo ad individuare le casistiche non standard che il programma non gestisce, come l'inserimento di caratteri anziché di numeri nelle operazioni numeriche che dovremo andare a "controllare" oppure il controllo sull'inserimento di più di 10 caratteri nella stringa.

Nel codice correggeremo anche gli errori logici e di sintassi inserendoli come commenti con "//".

Andremo ora a correggere e proporre soluzioni nel codice.

```
#include <stdio.h>

void menu ();                   //Le funzioni void sono funzioni che non restituiscono un valore di ritorno
void moltiplica ();             //Non hanno bisgno quindi dell'istruzione return
void dividi ();
void ins_string();

int main ()

{
        int a,b; //inserico i valori
        char scelta = '\0'; //Ho levato le parentesi {}
        menu ();
        do{     //inserisco un ciclio do-while per controllare gli input

        scanf ("%c", &scelta);          //Ho inserito "%c" al posto di "%d" per l'inserimento di un carattere anzichè solamente un decimale

        switch (scelta)
        {
                case 'A':
                        moltiplica();
                        break;
                case 'B':
                        dividi();
                        break;
                case 'C':
                        ins_string();
                        break;
                case 'D':  //in caso si volesse chiudere il programma senza usare Ctrl+Z per chiuderlo
                        return 0;
                default:
                        printf("\n\nInserisci A, B, C o D\n\n"); //aggiungo un default
                        rewind(stdin);
                        break;
                }
        }while(scelta != 'A' || scelta != 'B' || scelta != 'C');  //se si sceglierà un carattere diverso da questi ci darà un errore

return 0;
}
```

Questo sarà il primo blocco del codice corretto prima delle funzioni void.

Ho inserito dei cicli do-while che permette di eseguire un blocco di codice mentre risulta vera una determinata condizione, eseguendo almeno una volta le istruzioni

```c
void menu ()
{
        printf ("\nBenvenuto, sono un assitente digitale, posso aiutarti a sbrigare alcuni compiti\n"); //aggiungo degli \n per inserire spazi per dare pulizia
        printf ("\nCome posso aiutarti?\n");
        printf ("A >> Moltiplicare due numeri\nB >> Dividere due numeri\nC >> Inserire una stringa\nD >> Esci dal programma\n");
}

void moltiplica ()
{

        int a; //tolgo short int e metto solo int per inserire i numeri
        int b;
        int prodotto; //aggiungo l'assegnazione del prodotto qui
        int controllo_a;  //inserisco l'assegnazione dei controlli
        int controllo_b;
         printf ("Inserisci i due numeri da moltiplicare:\n\n");
do{  //inserisco un ciclo do-while
        printf("Inserisci il primo numero: ");
       controllo_a =  scanf("%d", &a);               //questo scanf deve essere "%d" perchè è un numero intero

        printf("Inserisci il secondo numero: ");
       controllo_b =  scanf("%d", &b);

        if(controllo_a == 0 || controllo_b ==0){
                printf("Carattere non valido, inserisci un numero\n");
                rewind(stdin);
                break;
        }
        else{
        prodotto = a * b;
        printf ("\n\nIl prodotto tra %d e %d e': %d", a,b,prodotto); //inserisco \n per dare pulizia al programma
        }
        }while(controllo_a == 0 || controllo_b == 0);
}
```

```c
void dividi ()
{
        int a;
        int b; //come sopra separo l'assegnazione dei valori a e b
        int divisione; //inserisco l'assegnazione della divisione
        int controllo_a;
        int controllo_b;
do{  //uso sempre il ciclo do-while per il controllo
        printf ("Inserisci il numeratore:");
        controllo_a =  scanf ("%d", &a);

        printf ("Inserisci il denumeratore:");
        controllo_b = scanf ("%d", &b);

        if (b == 0){  //qui utilizzo un ciclio if-else per evitare che il denumeratore sia 0
                printf("\nil denumeratore non può essere 0\n");
                dividi();}
        else{
        divisione = a / b;  //ho messo l'operatore / (divisione) al posto dell'operatore % (modulo) per ottenere una divisione con risultato decimale

        printf ("\n\nLa divisione tra %d e %d e': %d", a,b,divisione);
        }
        }while(controllo_a == 0 || controllo_b == 0);
}



void ins_string ()
{
        char stringa[10];
        printf ("Inserisci la stringa:");
        rewind(stdin); //ripulisco il buffer
        fgets(stringa, 10, stdin);  //per non far superare il numero di caratteri della stringa
        printf(stringa);
}
```

Questi saranno i blocchi successivi con le funzioni void corrette. Nei quali, oltre ai cicli do-while sopra citati, ho inserito anche degli if-else che è un'istruzione di controllo condizionale utilizzato per eseguire una porzione di codice se si verifica una determinata condizione.

Andremo ora a verificare il corretto funzionamento del codice:


┌──(kali㉿kali)-[~]
└─$ ./EsercizioX
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
D >> Esci dal programma
A
Inserisci i due numeri da moltiplicare:
Inserisci il primo numero: 4
Inserisci il secondo numero: 5
Il prodotto tra 4 e 5 e': 20
Inserisci A, B, C o D
B
Inserisci il numeratore:8
Inserisci il denumeratore:2
La divisione tra 8 e 2 e': 4
Inserisci A, B, C o D
C
Inserisci la stringa:
OKOK
Inserisci A, B, C o D
D
┌──(kali㉿kali)-[~]
└─$