ESERCIZIO BUG HUNTING

1) Capire cosa fa il programma senza eseguirlo:

il programma inizialmente mette in condizione il processore di accedere alla libreria <c stdio.h> (dove ad esempio sono contenute le istruzioni scanf, printf ecc.), poi ci elenca le diverse funzioni che dovranno esser lette durante il processo, specificando con void che la funzione specificata non accetta altri parametri .
In questo caso le funzioni da utilizzare sono void menu,void moltiplica, void dividi,void ins_string).
Le () stanno ad indicare che le funzioni che le precedono non necessitano di un input da parte dell utente.
Int main () sarà da dove il processore comincera a leggere per far partire il programma:
Con la parentesi graffa, diamo inizio al blocco di codice che dovra esser letto (che chiuderemo con la graffa chiusa).
Nello specifico, in questo primo blocco di codice assegnamo al nuovo blocco di memoria "scelta" la variabile char da 1 byte che puo contenere un numero da 1 a 9 e qualsiasi lettera ma puo contenere un solo carattere .
Chiediamo all utente di scegliere una di queste tre opzioni (a,b,c). Con l'itruzione scanf leggo e chiedo un input all utente che verra salvato nel parametro "scelta" (quindi gli chiedero di scegliere una lettera che verra salvata grazie a questa istruzione), poi mediante lo switch, che e una istruzione che permette, in base al cambiamento di variabile (char scelta) quale blocco di dati incanalare, sceglierò cosa fare: quindi aprirò un nuovo blocco di codice e se sceglierò A deciderò di fare una moltiplicazione (ed incanalare su blocco dati associato alla moltiplicazione, (void moltiplica)), con B una divisione(void dividi), con c l' inserimento di una stringa (void ins_stringa). Con il "BREAK" termino il compito dello switch dopo una determinata scelta dell'utente (quindi come abbiamo detto a:moltiplicazione, b: divisione c:inserimento stringa).
Chiudiamo il blocco di dati "switch" e usiamo la funzione return 0, che usato nella funzione main determina la chiusura del programma (return 0 restituisce 0 e indicherà che il programma e stato chiuso correttamente.) ed infine chiudiamo con la graffa anche il programma main.

Ora apriamo il blocco di memoria salvato come void menu e, tramite printf, che si occupa di fare da ponte fra la scrittura di programmazione e l interfaccia grafica dell utente, chiediamo all utente (che vedra sullo schermo vero e proprio cio che e scritto esattamente printf ("QUI");) quale delle tre opzioni scegliere.
Se si sceglierà l opzione A:
la lettura del programma verrà incanalata verso il blocco di dati corrispondente, perciò void moltiplica():
salverò i numeri scelti dall'utente (a,b) nel blocco memoria short int (short variabile da 2 byte numeri interi), quindi tramite l istruzione printf chiederò all'utente di scegliere due numeri da moltiplicare: con scanf leggerò i due input dell'utente e salverò il primo come "a" (all' interno della istruzione scanf, il valore ricevuto con l'input richiesto dall'utente (%d) deve essere

salvato come a (&a), ricordando che ogni lettera dopo la % e univoca per il tipo di variabile che stiamo usando, quindi d : int, f : float, c : char ecc.).
Nel blocco di mermoria salvato come “prodotto”, salverò il prodotto fra i due numeri scelti dall’utente(a,b) moltiplicandoli fra di loro (*).
Infine riuserò printf per far apparire sullo schermo dell’utente il risultato:
printf (“il prodotto tra %d e %d è %d”, a,b,prodotto) specificando che il primo valore %d debba essere il valore che abbiamo salvato in short int come “a” nella istruzione printf, mentre il secondo valore %d debba essere il valore salvato come “b”, infine il terzo valore debba essere il prodotto fra “a” e “b” che abbiamo salvato nell blocco memoria short int prodotto (dove gli abbiamo detto di essere per l’appunto la moltiplicazione di “a” e “b”: a*b).
Ed infine chiudiamo il blocco dati associato a void moltiplica, dato che abbiamo dato all’utente la possibilita di moltiplicare due numeri e vedere il risultato, quindi il nostro obiettivo è stato raggiunto.
Prendiamo in considerazione l’ipotesi che l’utente scelga l’opzione B e, come abbiamo detto, grazie all istruzione switch, gli si apra il blocco dati memorizzato nello slot void dividi:
Il blocco dati in questione si comporterà e seguirà le stesse dinamiche salvate in void moltiplica: salverà i due valori “a” e “b” in int a,b e, mediante la funzione printf chiedera “graficamente” all’utente di scegliere due valori da assegnare ad “a” e “b” tramite l istruzione scanf,
infine, diverso dalla moltiplicazione, assegnerà allo slot int divisone il risultato della divisione fra i due valori scelti dall’utente “a” e “b” (a %b) dove la % sta ad indicare nel linguaggio C la divisione. Dopo, tramite printf, farà comparire sull’interfaccia grafica dell’utente il risultato della divisione: printf(“il risultato della divisione fra %d e %d è %d”,a,b,divisione) dove “a” sarà il primo valore scelto dall’utente, “b” il secondo e “divisione” il risultato della divisone dei due valori scelti dall’utente salvato nello slot int divisione : a % b.

Ora prendiamo in ipotesi l’ultimo caso in cui l’utente scelga l’opzione C:
Si incanalerà il blocco dati salvato nello slot void ins_string dove l’utente sceglierà una frase (da massimo 10 caratteri come indicato dal [10]) da salvare nello slot char stringa.
Quindi l’utente chiederà “graficamente” all’utente, tramite printf, di inserire una stringa, l’input dell’utente lo raccoglieremo mediante la funzione scanf che salverà la frase (%s) nello slot char stringa (&stringa) .
QUINDI PER RICAPITOLARE, QUESTO PROGRAMMA, UNA VOLTA AVVIATO, DARà ALL’UTENTE TRE OPZIONI DA SCEGLIERE TRAMITE UN INPUT DALLA TASTIERA :
A)MOLTIPLICARE DUE FATTORI SCELTI DA LUI ED OTTENERE IL RISULTATO
B)DIVIDERE DUE FATTORI SCELTI DA LUI ED OTTENERE IL RISULTATO
C)SCEGLIERE UNA FRASE E VEDERLA SCRITTA

2) Individuare dal codice sorgente le casistiche non standard che il programma non gestisce

Nel codice sorgente abbiamo usato il tipo di funzione “void”, che a differenza ad esempio del tipo “int”, non richiede alcun codice o valore di ritorno (difatti non ha bisogno della funzione “return”) che ad esempio invece abbiamo trovato nel blocco int main (return0 : la procedura è stata eseguita con successo).

In sintesi ed in maniera pratica possiamo dire che void non si potrebbe usare nel blocco di codice (int main) dove si attiva la funzione dello switch, perché? Perché li la scelta dell'utente (tramite input) deve per forza tornare alla funzione per far partire la determinata cosa che vuole fare (se moltiplicare dividere o scrivere una frase), mentre, nelle funzioni void che troviamo sotto di moltiplicazione divisione o creazione di una frase, non ho bisogno del return dato che l opzione che volevo dare all'utente è terminata e non ha bisogno di alcun valore di ritorno.

3) Individuare eventuali errori di sintassi e logici ed in caso come risolverli

1)Il primo errore che si trova è nella funzione int main: la variabile assegnata per la variabile char in scanf, dovrebbe essere %c anziché %d che è per int.
Soluzione : cambiare
Scanf ("d", &scelta) con
Scanf ("c",&scelta)

2)nell'istruzione switch conviene mettere il case "default" in caso nessuno dei 3 case (a,b o c) avvenga,per sbaglio dell'utente o per altri motivi, e vieni rimandato all'inizio del processo.
Soluzione: aggiungere:

```
default: != 'a', 'b', 'c'
printf ("devi scegliere una lettera tra 'a','b','c' ");
break;
```

3)Un altro errore si trova nel comando scanf all'interno della funzione void moltiplica, difatti viene usata una sintassi sbagliata per la variabile di tipo int, che vuole per la precisazione %d anziche %f (float).
Soluzione: cambiare
Scanf ("f",&a) con
Scanf ("d",&a)

4)un altro errore è nella funzione void dividi:
a "divisione" bisogna assegnare una variabile "float", dato che il risultato potrebbe venire anche con la virgola.
Soluzione:
sostituire
"int divisione = a % b"
con "float divisione = a % b"
e, di conseguenza, sostituire
printf ("la divisione fra %d e %d è %d", a, b, divisione)
con printf ("la divisione fra %d e %d è %f", a, b, divissione);