Il programma prevede, tramite l'utilizzo di un menù con scelte preimpostate(A,B,C),le funzionalità di un calcolatore attraverso le scelte A(moltiplicazione) e B(divisione), invece selezionando l'opzione C ci fornirà la possibilità di memorizzare in un vettore di caratteri una determinata stringa scelta dall'utente.

ERRORI PRESENTI NEL CODICE

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

```
char scelta;
menu();
scanf("%c", &scelta);
```

Quando dichiaro la variabile scelta come
carattere, non ho la necessità di assegnarli il carattere terminatore '\0' in quanto non funzionale ai fini del programma; il carattere terminatore invece risulta essenziale nelle stringhe, infatti quando scelgo la dimensione del vettore di caratteri devo tener conto della presenza del carattere terminatore. Un'altro errore è individuato nella scanf, essendo un carattere la scelta corrette è sostituire %d(da utilizzare con gli interi) con %c.

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

```
switch(scelta){
    case 'A':
       moltiplica();
       break;
    case 'B':
       dividi();
       break;
    case 'C':
       ins_string();
       break;
    default:
       printf("errore:scelta non valida\n");
       break;
}
```

All'interno dello switch possiamo notare che il codice non prevede il caso 'default', ovvero non tiene conto della possibilità in cui l'utente seleziona un'opzione non presente tra 3 illustrate.

```
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 moltiplica(){

      short int a,b,prodotto;
      printf("inserisci i due nuemri da moltiplicare:\n");
      scanf ("%hd",&a);
      scanf("%hd",&b);

      prodotto=a*b;
      printf("il prodotto tra %hd e %hd è: %hd\n",a,b,prodotto);
}
```

Una volta chiamata la funzione 'moltiplica', nella prima scanf %f fa riferimento ai numeri reali ma la variabile 'a' è dichiarata come short int, in questo caso caso sostituiamo con %hd. Notare che short int occupa 2 byte, ovvero 16 bit, ha quindi un intervallo di valori che va da 0 a circa 65000.

```c
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);
}
```

```c
void dividi (){
      int a,b;
      float divisione;
      printf("inserisci il numeratore:\n");
      scanf("%d",&a);
      printf("inserisci il numeratore:\n");
      scanf("%d",&b);

      divisione=(float)a/b;
      printf("la divisione tra %d e %d è: %.2f\n",a,b,divisione);
}
```

a%b ci restituisce il resto della divisione e non il risultato.
Dichiarando la variabile 'divisione' come int non avrò il risultato esatto della divisione, in questo caso è più corretto ricorrere all'utilizzo dei numeri reali dichiarando 'divisione' come un float, affinchè abbia senso l'operazione è necessario effettuare un cast o sul numeratore o sul denominatore.
Nella printf finale sostituirò il %d relativo alla divisione con il %.2f affinchè mi stampi a video il valore reale con al massimo due cifre dopo la virgola.

```c
void ins_string ()
{
        char stringa[10];
        printf ("Inserisci la stringa:");
        scanf ("%s", &stringa);
}
```

```c
void ins_string(){
      char stringa[10];
      printf("inserisci la stringa\n");
      scanf("%s",stringa);
}
```

Nella scanf non è necessario utilizzare '&', questo perchè stringa=&stringa[0], ovvero punta già all'indirizzo di memoria della prima cella occupata dal vettore di caratteri.

CASISTICHE NON STANDARD

```c
#include <stdio.h>
#include <ctype.h>
#include <string.h>
```

```c
int main(){
    char scelta;
    menu();
    scanf("%c", &scelta);
    //controllo che il carattere inserito sia alfabetico
    if(isalpha(scelta)){
        //anche se l'utente inserisce il carattere minusculo lo converto al maiuscolo
        scelta=toupper(scelta);
        switch(scelta){
            case 'A':
                moltiplica();
                break;
            case 'B':
                dividi();
                break;
            case 'C':
                ins_string();
                break;
            default:
                printf("errore:scelta non valida\n");
                break;
        }
    } else
        printf("errore:carattere non alfabetico");
    return 0;
}

void menu(){
```

Nella funzione main oltre che inserire il default come ho illustrato precedentemente, ho dichiarato due nuove librerie, una per i caratteri e l'altra per le stringhe , in modo tale da poter utilizzare le funzioni non presenti in libreria , infatti nel codice ho aggiunto la verifica al carattere inserito, isalpha è una funzione che restituisce un valore diverso da 0 quando il carattere analizzato è alfabetico, nel caso in cui dovesse ritornare il valore 0 il ciclo continua nell'else dove mi stamperà a video l'errore.
Se l'if da un riscontro positivo, attraverso l'utilizzo della funzione toupper modifico in maiuscolo il carattere inserito in modo da non avere problemi di selezione nello switch.

Una volta partito lo switch, verificherà la natura del carattere per vedere se combacia con le opzioni disponibili fornendo un errore nel caso in cui la scelta dell'utente non dovesse rientrare nei vari case dello switch.
Se dovesse trovare riscontro in uno dei case elencati, la funzione main richiamerà le varie funzioni dichiarate all'inizio del programma ed eseguirà la richiesta dell'utente.

```
#define N 10
```

```
void ins_string(){
      char stringa[N];
      int lung,fine;
      //ciclo do while, nel caso in cui l'utente inserisce una stringa con dei parametri non validi ha la possibilità
      //di reinserire la stringa
      do{
          //inserisco un flag per uscire dal ciclo, lo imposto sull'uscita(0) in modo che continui il ciclo solo quando  riscontra un errore(1)
          fine=0;
          printf("inserisci la stringa con al massimo 10 caratteri:\n");
          //prediligo l'utilizzo della gets allo scanf perchè all'interno della stringa memorizza anche eventuali spazi e a capo
          //che sostituisce come carattere terminatore
          //scanf memorizza una sequenza di caratteri fino allo spazio o a capo, quindi solo parole, gets invece memorizza anche frasi
          gets(stringa);
          //effettuo un controllo sulla stringa per verificare che la sua lunghezza rientri nella dimensione fornita al vettore
          lung=strlen(stringa);
          if(lung>N-1){
               //ripete il ciclo
               fine=1;
               printf("errore: lunghezza stringa eccessiva\n");
          }
          else
               printf("la stringa inserita è: %s\n",stringa);
      } while (!fine)
}
```

Inizialmente attraverso una #define dichiaro che il valore di N è 10, questa sarà la dimensione effettiva del vettore stringa compresa del carattere terminatore '\0'; ho deciso di implementare il codice dapprima sostituendo lo scanf con la gets in modo da consentire all'utente di poter inserire più di una parola, dopo effettuo un controllo sulla lunghezza della stringa con la funzione strlen grazie alla libreria string.h.
Ho optato per un ciclo do while in modo tale da consentire all'utente di reinserire la stringa se si dovesse verificare un errore.
Il ciclo if verifica se la lunghezza della stringa supera quella della dimensione prevista dal vettore, in caso di riscontro positivo mi stamperà a video l'errore e mi cambierà il valore del flag di uscita in modo tale da poter far iniziare nuovamente il ciclo do, altrimenti mi stampa a video la stringa inserita dall'utente ed esce dal prima dal ciclo e successivamente dalla funzione.