File di Testo (ASCII) – Matrici – Strutture: BATTAGLIA NAVALE

battComplNavi.txt

```
0 0
4 8
6 5
9 8
2 6 2 7
2 0 3 0
8 5 8 6
0 7 0 8 0 9
2 2 2 3 2 4
6 3 7 3 8 3 9 3
```

Il programma legge da un file di testo ("*battComplNavi.txt*") le coordinate delle 10 navi (4 navi da 1 cella, 3 navi da 2 celle, 2 navi da 3, 1 nave da 4) da disegnare nella mappa(10righex10colonne); chiede quindi la digitazione del ***nome del file*** dei colpi (cioè le coordinate delle posizioni indicate dall'avversario) e per ogni coppia di coordinate fornisce a video l'esito del colpo (acqua, colpito). Infine, riproduce la mappa con le navi (una X indica in quali posizioni sono colpite) e l'esito della partita (colpi effettuati e punti totalizzati: un punto per ogni colpo andato a segno). *NOTA: la mappa delle navi viene prodotta a video anche all'inizio dell'esecuzione; l'angolo in alto a sin, ha coordinate (0,0), in basso a destra (9.9).*

esecuzione programma (parte 1)

```
Prompt dei comandi - ...
>>battagliaNavAffCompl.exe


mappa
1 . . . . . 3 3 3
. . . . . .
2 . 3 3 3 . 2 2 . .
2 . . . . . .
. . . . . 1 .
. . . . . .
. . . 4 . 1 . . .
. . . 4 . . . . .
. . . 4 . 2 2 . . .
. . . 4 . . . . 1 .


digita il nome del file COLPI
(senza estensione): battComplColpi
```

battComplColpi.txt

```
0 0
9 9
0 8
1 8
0 9
0 7
2 0
2 1
3 0
2 2
2 3
2 4
4 9
4 8
9 8
9 3
9 4
8 3
7 3
6 3
2 0
6 5
8 5
8 6
2 6
2 7
```

esecuzione programma (parte 2)

```
Prompt dei comandi

 aperto il file:battComplColpi.txt

0 0 nave da 1 colpita e affondata
9 9 acqua
0 8 nave da 3 colpita
1 8 acqua
0 9 nave da 3 colpita
0 7 nave da 3 colpita e affondata
2 0 nave da 2 colpita
2 1 acqua
3 0 nave da 2 colpita e affondata
2 2 nave da 3 colpita
2 3 nave da 3 colpita
2 4 nave da 3 colpita e affondata
4 9 acqua
4 8 nave da 1 colpita e affondata
9 8 nave da 1 colpita e affondata
9 3 nave da 4 colpita
9 4 acqua
8 3 nave da 4 colpita
7 3 nave da 4 colpita
6 3 nave da 4 colpita e affondata
2 0 punto gia' colpito
6 5 nave da 1 colpita e affondata
8 5 nave da 2 colpita
8 6 nave da 2 colpita e affondata
2 6 nave da 2 colpita
2 7 nave da 2 colpita e affondata

colpi=26        punti=20

hai colpito tutte le navi:
COMPLIMENTI !!!
mappa
X . . . . . X X X
. . . . . 0 .
X 0 X X X . X X . .
X . . . . . .
. . . . . X 0
. . . . . .
. . . X . X . . . .
. . . X . . . . . .
. . . X . X X . . .
. . . X 0 . . . X 0

>>
```

programma (parte 1)

```
#include <iostream>
using namespace std;
#include <string.h>
#include <fstream>
#define SP   ' '
#define TAB  '\t'
#define N    10
#define COLP 'X'
#define NOCP '.'
#define AQUA '0'

struct Posiz {
   int x; int y;      //--- coordinate di 1 cella di nave
};

struct { //--- struttura senza nome utilizzata solo per nave
  int    tipo;            //--- 1/2/3/4 (nave da1/2..)
  Posiz  posz[4];  //--- tabella di struttura Posiz di 4 elementi
  char   colp[4];       //--- '.' (non colpito)  'X' (colpito)
  int    NC;          //--- coordinate colpite 0/1/2/3/4
} nave[10];       //--- tabella di struttura di 10 elementi

 int mappa[10][10] = {0}, iNv=0,
     x, y, i, j, tipo;    //--- iNv progressivo nave

 fstream fnavi, fgame;        //--- file ASCII di input
```

programma (parte 2)

```
void insNavi(){
      nave[iNv].tipo          = tipo;
      nave[iNv].NC            = 0;
      for (j=0; j<tipo; j++){
        fnavi >> x >> y;
        mappa[x][y]           = tipo;
        nave[iNv].posz[j].x   = x;
        nave[iNv].posz[j].y   = y;
        nave[iNv].colp[j]     = NOCP;
      }
      iNv++;
}

          //-----inizializzazione matrice
void inizMappa (void){
    for (i=0; i < 4; i++){
        tipo = 1;
        insNavi();
    }
    for (i=0; i < 3; i++){   //--- 3 navi da 2
        tipo = 2;
        insNavi();
    }
    for (i=0; i < 2; i++){   //--- 2 navi da 3
        tipo = 3;
        insNavi();
    }
      tipo = 4;              //--- 1 nave da 4
      insNavi();

    fnavi.close();
}

void avideoMappa (void){
     cout<< "\n\nmappa\n";
     for (int i=0; i < N ; i++) {
      for (int j=0; j < N; j++)
          if (mappa[i][j]==0)
              cout << NOCP << SP;
          else
          if (mappa[i][j]==9)
              cout << COLP << SP;
          else
          if (mappa[i][j]==8)
              cout << AQUA << SP;
          else
              cout << mappa[i][j] << SP;
      cout << endl;
     }
     cout << endl;
}
```

DESCRIZIONE programma

La tabella **nave** (array di strutture di 10 elementi) contiene la tabella **posz** di tipo Posiz di 4 elementi: per le navi da 1 (**tipo = 1**) viene valorizzato solo l'elemento di posto 0 di **posz**, per quelle da 2, gli elementi di posto 0 e 1, per la nave da 4 vengono valorizzati tutti gli elementi (di posto 0,1,2,3) a partire dalle coordinate **x** e **y** lette dal file **fnavi**.

La tabella **nave** contiene anche l'array di char **colp** di 4 elementi (*parallelo* dell'array **posz**) che viene utilizzato per memorizzare lo stato (colpito o non colpito) del segmento di nave corrispondente (colp[0] indica lo stato di posz[0], colp[1] lo stato di posz[1] etc.).

programma (parte 3)

```
//--- verifica se le coordinate fornite sono di una nave
//- restituisce L e aggiorna 2 parametri (pass. x riferimento)
int cercaColpi (int x, int y, int T,
                          char& S, int& j){
  int   L;  char trovato = 'N';
    if (T==2) L=4;   //--- ricerca segmenti da 4^nave
    else
    if (T==3) L=7;   //--- ricerca segm. dalla 7^ nave
    else
    if (T==4) L=9;    //--- ricerca segm. ultima nave

    while (L < iNv && trovato == 'N'){
      j=0;
      while (j<T && trovato == 'N'){
        if (nave[L].posz[j].x == x
        && nave[L].posz[j].y == y){
            trovato = 'S';
            S = nave[L].colp[j];
        } else
            j++;  //--- incremento segmento nave
      }
      if (trovato == 'N')
          L++;    //--- incremento progressivo nave
    }
    return L;       //--- progressivo nave
}
int main() {
 int x, y, punti=0, colpi=0, K, tipo, j;
 char stato;
 char nomefile[30];
//--- costante per accodare l'estensione del file
 const char est[5]=".txt";

  fnavi.open ("battComplNavi.txt",  ios::in);
     if (fnavi.fail() ){
        cout << "FILE navi NON TROVATO\n\n";
       return -900;
     }
     inizMappa ();
     avideoMappa ();
```

programma (parte 4)

```
      cout << "\ndigita il nome del file COLPI";
      cout << "\n(senza estensione): ";
      cin  >> nomefile;
      strcat (nomefile, est);          //--- costruzione del nomefile completo mediante concatenazione
      fgame.open  (nomefile, ios::in);       //il nome esterno del file è fornito mediante variabile
      if  ( fgame.fail()){
          cout << "FILE colpi NON TROVATO\n\n";
        return -900;
      }
      cout << "\n aperto il file:"<< nomefile << "\n\n";

//-------- ciclo di simulazione il gioco ---------

     fgame >> x >> y;                                  //--- lettura fuori ciclo ----
     while ( ! fgame.eof() ){
     colpi++;
     if (mappa[x][y] == 9 || mappa[x][y] == 8)
             cout<<x<<SP<<y<<SP<<"punto gia' colpito"<<endl;
     else
     if (mappa[x][y] == 0){
             cout<<x<<SP<<y<<SP<<"acqua\n";
         mappa[x][y] = 8;                              //--- colpita coordinata di tipo ACQUA ---
     }
     else {
          if (mappa[x][y] == 1) {                      //--- colpita nave da 1 (e affondata) ---
              cout<<x<<SP<<y<<SP<<"nave da 1 colpita e affondata"<<endl;
              punti++;
          }
          else {                                       //--- colpita nave da 2,3 o 4 -------
          tipo = mappa[x][y];
          K = cercaColpi ( x, y, tipo, stato, j);      //--- controlla rese colpiti tutti i segmenti ---
          if (stato == COLP)                           //--- stato e j determinati da cercaColpi
              cout<<x<<SP<<y<<SP<<"nave da  "<<mappa[x][y]<<" gia' colpita"<<endl;
          else {
              nave[K].colp[j] = COLP;
              nave[K].NC++;
              cout<<x<<SP<<y<<SP<<"nave da  "<<tipo<<" colpita";
              if (nave[K].NC == mappa[x][y]) //---colpiti tutti i segmenti della nave=affondata) --
                  cout<<" e affondata";
              cout<<endl;
              punti++;
          }
       }
       mappa[x][y] = 9;                                //--- colpita coordinata di tipo NAVE ---
       }
     fgame >> x >> y;                                  //--- lettura nel ciclo ----
     }                                                 //--- FINE del ciclo while ----
     cout << "\ncolpi="<< colpi<<'\t'<<"punti="<< punti<<"\n\n";
     if (punti == 20)
         cout<<"hai colpito tutte le navi:\nCOMPLIMENTI !!!\n";
     else
          cout<<"partita NON completata\n";
     avideoMappa ();
}
```