Di seguito viene riportato lo svolgimento di una simulazione d'esame, con l'implementazione guidata svolta in aula e la relativa discussione in forma di commenti.

TIRO AL BERSAGLIO

Il testo della prova d'esame è disponibile sulla piattaforma Kiro, sotto le cartelle:
per gli iscritti di Bioingegneria
`FondInf-bioing > Tutorato > esame bersagli > bersaglio.pdf`
per gli Eletronici-Informatici
`FdI 2016-17 > Tutorato 12 - "Bersaglio" > Traccia tutorato`.

BERSAGLIO2.C

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

#define DIM_NOME 120 //buffer di dim ragionevole
#define DIM_BUFF 120
#define DIM_PUNTI 15 //numero di tiri di ogni giocatore

struct giocatore {
  char nome[DIM_NOME];
  int dist[DIM_PUNTI];
    int punti_tot;
};

// diametri dei cerchi
int dist[] = { 60, 100, 140, 180, 220, 260, 300 };
// punti per ogni cerchio (minore od uguale del diametro)
int punti[] = { 15, 10, 7, 5, 3, 2, 1 };

    // ---- PRIMA PARTE ---- //
struct giocatore *caricaFile(FILE *fin, int *tot){ [...] }
int *maxCentri(struct giocatore *giocatori, int tot, int *tot_bombers){[...]}
void dist_min_max(struct giocatore *g, int tot, int *dist_min, int *dist_max)
    { [...] }
    //---- FINE PRIMA PARTE ---- //

void dist_medie(struct giocatore *g, int tot, double medie[])
{
  int i, j;

  for (i=0;i<tot;i++) {
    medie[i] = 0.0;
    for (j=0;j<DIM_PUNTI;j++) {
      //sommo tutte le distanze del giocatore i-esimo
      medie[i] += g[i].dist[j];
    }
    medie[i] /= DIM_PUNTI; //faccio la media :)
  }
}

int compara(const void *g1, const void *g2)
{
  const struct giocatore *a = g1, *b = g2;
  /*
    Dati due giocatori g1 e g2, ritorna
```

```
45      la differenza di punti fra il secondo ed il primo,
46      ed usa questa come metodo di comparazione, ovvero:
47      se è >= di zero vuol dire che g2 "è maggiore" di g1,
48      quindi va messo prima in classifica
49     */
50    return b->punti_tot - a->punti_tot;
51  }
52
53  void classifica(struct giocatore *giocatori, int tot) {
54    /*
55        Ciclo su tutti i tiri di tutti i giocatori,
56          e per ogni tiro (distanza dentro giocatore.dist[]) controllo se
57          tale distanza è <= delle distanze predefinite in dist[]:
58          Tullio ha tirato a 140 mm, comparo 140 mm prima con dist[0], dist[1],
      dist[2] qui becco la condizione if vera, quindi aggiungo i punti
59      */
60
61    int i,j,k;
62      for(i=0; i < tot; i++){
63          giocatori[i].punti_tot =0;
64          for(j=0; j < DIM_PUNTI; j++){
65            for(k=0; k < 7; k++){
66              if(giocatori[i].dist[j]<= dist[k] / 2 ) {
67                giocatori[i].punti_tot += punti[k];
68                break;
69              }
70            }
71        }
72      }
73
74      // Ordino effettivamente tutti i giocatori nella parte di memoria loro
     allocata secondo il criterio definito dalla funzione "compara"
75      qsort(giocatori, tot, sizeof(struct giocatore), compara);
76
77  }
78
79  int main(int argc, char **argv) {
80
81      /* //debug
82      int i,j;
83      printf("#numero arg: %d\n",argc);
84      for(i=0;i<argc;i++)
85      printf("#argv[%d]: \"%s\"\n", i, argv[i]);
86          */
87
88    int tot; //tot giocatori
89    int tot_bombers =0; // tot campioni
90    FILE *fin; //ptr al file
91    struct giocatore *giocatori;
92
93    fin = fopen(argv[1],"r"); //apro file
94    giocatori = caricaFile(fin,&tot); //funzione carica
95    fclose(fin); //chiudo
96
97    /* //debug
98    for(i=0; i<tot; i++) {
99      printf("#%s#",giocatori[i].nome);
100     for(j=0;j<DIM_PUNTI;j++)
101       printf("%d ",giocatori[i].dist[j]);
102       printf("\n");
```

```
103   }
104    */
105
106   int *bombers; // i campioni con ugual punteggio
107   bombers = maxCentri(giocatori,tot, &tot_bombers);
108
109   int dist_min, dist_max;
110   dist_min_max(giocatori, tot, &dist_min, &dist_max);
111   printf("[MINMAX]\n%d\n%d\n", dist_min, dist_max);
112
113   double medie[tot];
114   dist_medie(giocatori, tot, medie);
115   printf("[MEDIA]\n");
116   for(i=0; i < tot; i++)
117     printf("%.1lf %s", medie[i], giocatori[i].nome);
118
119   classifica(giocatori, tot);
120   printf("[CLASSIFICA]\n");
121   for(i=0; i < tot; i++)
122     printf("%d %s", giocatori[i].punti_tot, giocatori[i].nome);
123
124   return 0;
125 }
```

Come nei casi precedenti, è utile verificare la soluzione con pvcheck fornito su Kiro:
`./pvcheck ./a.out bersagli.test`