Soluzione

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#define DIM 24

typedef struct{
		char nome[20];
		float x;
		float y;
} centro;

centro provFe[DIM]={
	{"Ferrara", 44.830551, 11.617699},
	{"Focomorto", 44.831528, 11.678467},
	{"Palmirano", 44.780617, 11.695290},
	{"Portomaggiore", 44.696968, 11.804466},
	{"Molinella", 44.617355, 11.668510},
	{"Cento", 44.728931, 11.289482},
```

```
        {"Bondeno", 44.887257, 11.420975},
        {"Mirandola", 44.886524, 11.063919},
        {"Cenacchio", 44.694286, 11.430931},
        {"Minerbio", 44.623955, 11.490326},
        {"Consandolo", 44.653023, 11.775627},
        {"Argenta", 44.614666, 11.833305},
        {"Filo", 44.587044, 11.929092},
        {"Comacchio", 44.692577, 12.182808},
        {"Lagosanto", 44.761604, 12.139549},
        {"Codigoro", 44.828846, 12.107964},
        {"Mesola", 44.919842, 12.228470},
        {"Alberazzo", 44.888229, 12.250099},
        {"Goro", 44.853191, 12.298508},
        {"Donzella", 44.930050, 12.328033},
        {"Copparo", 44.891392, 11.825409},
        {"Tamara", 44.875580, 11.771851},
        {"Pontelagoscuro", 44.874119, 11.606712},
        {"Spinazzino", 44.709904, 11.630745}

};
```

```
main(){
	int i, j, coppia[2], trovato1=0, trovato2=0;
	float dx, dy, distanzamin, distanza;
	char primo[20], secondo[20];

	/* 1^ */
	for(i=0; i<DIM; i++){

		printf("centro %d: %s %f %f\n", i+1, provFe[i].nome,
			provFe[i].x, provFe[i].y);

	}

	/* 2^ */
	printf("inserisci il nome del primo centro: ");
	scanf("%s", primo);
	printf("inserisci il nome del secondo centro: ");
	scanf("%s", secondo);

	i=0;
```

```
while(i<DIM&&(!trovato1 || !trovato2)){

	if(!strcmp(primo, provFe[i].nome)){
		trovato1=1;
		coppia[0]=i;
	}
	if(!strcmp(secondo, provFe[i].nome)){
		trovato2=1;
		coppia[1]=i;
	}
	i++;
}

if(!trovato1){
	printf("impossibile trovare il primo centro\n");
}
if(!trovato2){
	printf("impossibile trovare il secondo centro\n");
}
if(trovato1 && trovato2){
```

```
        dx=provFe[coppia[0]].x-provFe[coppia[1]].x;

        dy=provFe[coppia[0]].y-provFe[coppia[1]].y;

        distanza=sqrt(dx*dx+dy*dy);
        printf("%s e %s distano %f\n", provFe[coppia[0]].nome,
              provFe[coppia[1]].nome, distanza);

}



/* 3^ */
/* se il primo centro non e` nella lista, troviamo la distanza del
   centro piu` vicino a Fe */
if(!trovato1){
        coppia[0]=0;
}
```

```
/* supponiamo che 100 sia sicuramente maggiore del minimo */
distanzamin=100;

for(i=0; i<DIM; i++){
	if(i!=coppia[0]){
		dx=provFe[coppia[0]].x - provFe[i].x;
		dy=provFe[coppia[0]].y - provFe[i].y;

		distanza=sqrt(dx*dx + dy*dy);

		if(distanza<distanzamin){
			distanzamin=distanza;
			coppia[1]=i;
		}
	}
}
printf("il centro piu` vicino a %s e` %s a distanza %f\n",
   provFe[coppia[0]].nome, provFe[coppia[1]].nome, distanzamin);

/* 4^ */
```

```
        /* vedi sopra */
        distanzamin=100;
        for(i=0; i<DIM; i++){

                for(j=i+1; j<DIM; j++){
                        dx=provFe[i].x - provFe[j].x;
                        dy=provFe[i].y - provFe[j].y;

                        distanza=sqrt(dx*dx + dy*dy);
                        if(distanza<distanzamin){
                                distanzamin=distanza;
                                coppia[0]=i;
                                coppia[1]=j;
                        }

                }
        }
        printf("i centri piu` vicini sono %s e %s, situati a distanza %f\n",
           provFe[coppia[0]].nome, provFe[coppia[1]].nome, distanzamin);
}
```