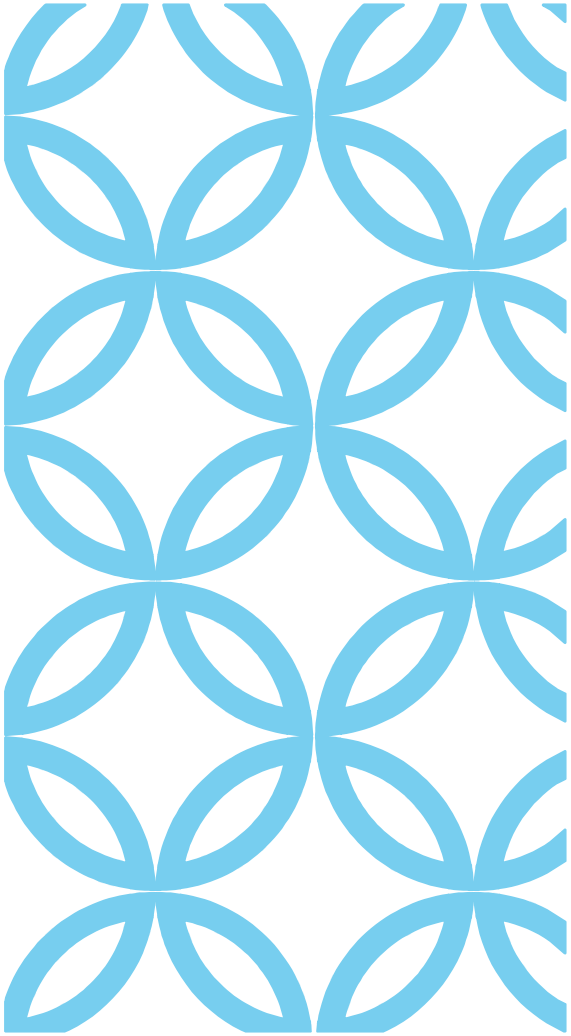

Corso di Programmazione e strutture dati
Docente di Laboratorio: Marco Romano
Email: marromano@unisa.it

ESERCITAZIONE: ABSTRACT DATA TYPES (ADT)

ADT: SPECIFICA

	Sintattica	Semantica
Tipi di dati	• Nome dell'ADT • Tipi da dati già usati	• Insieme dei valori
Operatori: Per ogni operatore	• Nome dell'operatore • Tipi di dati di input e di output	Funzione associata all'operatore • Precondizioni: definiscono quando l'oeratore è applicabile • Postcondizioni: definiscono relazioni tra dati di input e output

ADT: PUNTO

Sintattica	Semantica
Nome del tipo: Punto Tipi usati: Reale	Dominio: insieme delle coppie formate da due numeri reali: ascissa e ordinata
creaPunto (reale, reale) → punto	creaPunto(x, y) = p • pre: true • post: p = (x, y)
ascissa (punto) → reale	ascissa(p) = x • pre: true • post: p = (x, y)
ordinata (punto) → reale	ordinata(p) = y • pre: true • post: p = (x, y)
distanza (punto, punto) → reale	distanza(p1, p2) = d • pre: true • post: d = sqrt((ascissa(p1)-ascissa(p2))^2 + (ordinata(p1)-ordinata(p2))^2)

ESERCIZIO: PARTE 1-2

Estendere l'ADT Punto in modo da includere le seguenti funzionalità

1. Spostamento del punto dati due numeri reali deltaX e deltaY
2. Calcolo del centroide (posizione media) di un insieme di punti

SPOSTA

```
33  void sposta(Point p, float dx, float dy) {
34      p->x = p->x + dx;
35      p->y = p->y + dy;
36  }
```

CENTROIDE

```
39  Point centroide(Point sequenza[], int size) {
40      int i;
41      float sumx = 0.0, sumy = 0.0;
42
43      for(i = 0; i < size; i++) {
44          sumx += sequenza[i]->x;
45          sumy += sequenza[i]->y;
46      }
47
48      sumx /= size;
49      sumy /= size;
50
51      return createPoint(sumx, sumy);
52  }
```

ESERCIZIO: PARTE 3

Realizzare un programma che data una sequenza di punti

- Calcoli il numero di *m* coppie di punti che hanno distanza minore di un numero *d*

ANALISI – ESERCIZIO 3

Calcolare il numero di *m* coppie di punti che hanno distanza minore di un numero *d*

ANALISI – ESERCIZIO

- Dati di ingresso: sequenza s di n punti; Un numero reale d
  - Precondizione: n >= 2; d>=0;
- Dati di uscita: Intero m
  - Postcondizione: m è il numero di coppie di punti p1 e p2 in s tali che distanza(p1, p2) < d

Dizionario dei dati

Identificatore	Tipo	Descrizione
s	sequenza	sequenza di punti in input
n	intero	numero di elementi nella sequenza
d	reale	distanza massima tra una coppia
m	intero	numero di punti a distanza d
p1, p2	punto	punti tra cui valutare la distanza

PROGETTAZIONE – ESERCIZIO

1. Chiediamo il numero **n** di punti da aggiungere in un array
2. Creiamo un array **a** di punti di dimensione **n**
3. Chiediamo in input **n** punti caricandoli nell'array **a**
4. Chiediamo in input la distanza **d**
5. Calcoliamo il numero **m** di coppie in **a** con distanza minore di **d**
6. Stampiamo a video **m**

Attenzione: I passi 1, 2 4 e 6 sono direttamente implementabili con istruzioni nel programma principale

Realizziamo i sottoprogrammi per i passi 3 e 5

DISTANZA MINORE DI UN DATO NUMERO

```
58  int coppieDistMinD(Point sequenza[], int n, float
        dist) {
59      int i, j, cont = 0;
60      float d;
61
62      for(i = 0; i < n; i++) {
63          for(j = i + 1; j < n; j++) {
64              d = distanza(sequenza[i], sequenza[j]);
65              if(d <= dist) {
66                  cont++;
67              }
68          }
69      }
70
71      return cont;
72  }
```

MAIN...

```
10  int main() {
11      int scelta, n;
12      Point *s;
13
14      printf("Quanti nodi vuoi inserire? ");
15      scanf("%d", &n);
16      s= (Point*) calloc(n, sizeof(Point) );
17      riempiSequenza(s, n);
```

ESERCIZIO: PARTE 4

Realizzare un programma che data una sequenza di punti

- Calcoli la distanza massima fra le coppie di punti della sequenza

DISTANZA MASSIMA

```
74  float maxDistanza(Point sequenza[], int n) {
75      int i, j;
76      float max = 0, d;
77
78      for(i = 0; i < n; i++) {
79          for(j = i + 1; j < n; j++) {
80              d = distanza(sequenza[i], sequenza[j]);
81              if(d >= max) {
82                  max = d;
83              }
84          }
85      }
86
87      return max;
88
89  }
```

RIEMPIRE UNA SEQUENZA

```
51  void riempiSequenza(Point s[], int n) {
52      float x, y;
53
54      for(int i = 0; i < n; i++) {
55          printf("Inserire x e y.\n");
56          scanf("%f %f", &x, &y);
57
58          s[i] = createPoint(x, y);
59      }
60  }
```

DISTANZA FRA DUE PUNTI

```
28  float distanza(Point p1, Point p2) {
29      return sqrt(((p1->x - p2->x) * (p1->x - p2->x))
            + ((p1->y - p2->y) * (p1->y - p2->y)));
30  }
```