Tuples

Objectius:
- **Saber declarar una tupla i crear variables d'aquest tipus.**
- **Declarar una tupla que contingui altres tuples.**

(Objectius: 4.2.1 i 4.2.2)

(Documents relacionats: Estructures_C++.pdf)

1. Declaració de tuples

Observa el següent programa:

```
struct tMotor {
  string marca;
  int codi;
  float pes_total, volum_cilindre;
  int ncilindres;
  bool diesel;
};

int main()
{
  tMotor m1;
  tMotor m2 = { "Seat", 1013, 159.3, 350.0, 6, false };
  m1.marca = "Toyota";
  m1.codi = 401;
  m1.pes_total = 123.9;
  m1.volum_cilindre = 275.0;
  m1.ncilindres = 4;
  m1.diesel = true;
}
```

Aquest programa no fa res de bo, però compila, com a mínim. La declaració de la tupla, a dalt de tot, té 6 camps (membres de dades) de tipus diferents i defineix un tipus **`tMotor`**, que serveix per emmagatzemar certes dades d'un motor. El programa principal crea una variable **`m1`**, del tipus **`tMotor`**, i li assigna uns valors. També crea una altra variable **`m2`**, que inicialitza en el moment de declarar-la. Observa que pots, només veient el main, sense veure la declaració de la tupla, saber de quin tipus són els camps que té, quin nom tenen, i també el nom de la tupla.

A partir del següent programa, dedueix les tuples que s'haurien de declarar a dalt de tot per tal que compili correctament. Quan et falti informació, declara la tupla de la forma més lògica possible:

```
int main()
```

```
{
  tData d = { 27, 1, 1756 };

  tMesura m;
  m.poblacio = "Vic";
  m.temperatura = 15.9;

  tOrdinador o;
  o.cpu = "AMD Athlon 64";
  o.memoria_mb = 2048;
  o.velocitat = 3000;
  o.disc_dur_gb = 250.2;
  o.disquetera = false;
}
```

Substitueix en els 2 programes anteriors el codi per omplir la variable tupla per una acció/funció que faci aquesta tasca.

2. Persona

Dissenyeu un tipus de dades que permeti guardar les dades d'una persona: Nom, Cognoms, telèfon, e-mail, adreça.
Feu un programa que permeti omplir una variable d'aquest tipus i que visualitzi el seu contingut. Per fer això utilitzeu una acció/funció per omplir i un altre per visualitzar.

3. Distància punts

Dissenyeu un programa que donats dos punts en tres dimensions, calculi la distància que els separa. Feu servir tuples per emmagatzemar els punts.

4. Ordre de les declaracions

Quan es declaren estructures de dades de certa complexitat, és important posar-les en un ordre concret. Per exemple, les dues tuples

```
struct tVisita {
   string metge;
   tData dia;
};

struct tData {
   int dia, mes, any;
};
```

estan en l'ordre *incorrecte* perquè **`tData`** es declara més avall que **`tVisita`**, i resulta que **`tVisita`** fa servir **`tData`**. Per tant, la declaració d'una tupla ha de fer-se abans de utilitzar-la.

Dissenya un tipus per emmagatzemar un número de compte bancari. A continuació afegeix aquesta

dada a l'estructura Persona que has dissenyat abans i modifica el programa per que permeti omplir i visualitzar també aquest dada. Per fer això crea una acció/funció per introduir el compte i un altra per visualitzar-lo.