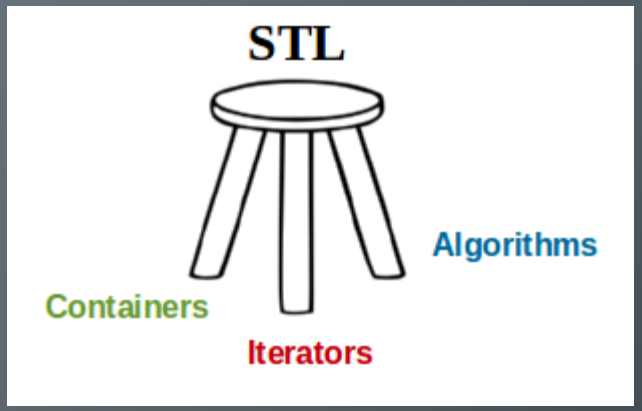
PROGRAMAÇÃO

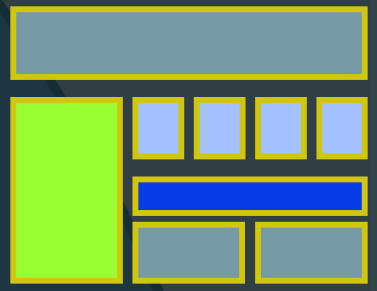
STL



Containers

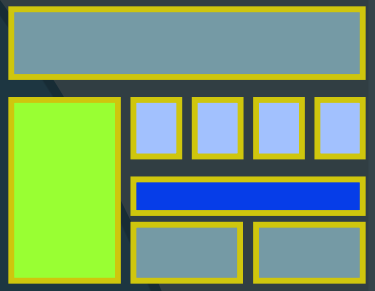
- **Containers Sequenciais:**
  - **Vector**
  - **Deque**
  - **List**
- **Containers Associativos:**
  - **Set**
  - **Map**
  - **Multiset & Multimap**
- **Adaptadores de Containers:**
  - **Stack**
  - **Queue**
  - **Priority Queue**

Estruturas



- Estruturas contém dados relacionados.
- Exemplos:
  - Registro de estudante: id, nome, gênero, ano de início, …
  - Conta no banco: número da conta, nome, saldo, …
  - Dados pessoais: nome, endereço, telefone, …
- Nas aplicações de banco de dados, estruturas são chamadas de registros.

Estrutura básica



- **Definição da estrutura:**

```
struct <struct-identifier>{
    <type> <identifier_list>;
    <type> <identifier_list>;
    ...
} ;
```

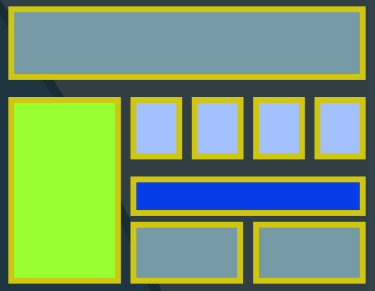
Cada identificador define um membro da estrutura.

- **ou:**

```
typedef struct {
    <type> <identifier_list>;
    <type> <identifier_list>;
    ...
} <struct-identifier>;
```

Outra forma de definir a estrutura

Estrutura básica



- **Exemplo:**

```
struct Data {
    int dia;
    int mes;
    int ano;
} ;
```

a estrutura “Data” tem 3 membros, dia, mes e ano.

- **Example:**

```
struct Estudante{
    string Nome;
    int Id;
    char Dept[5];
    char Genero;
};
```

A estrutura “Estudante” tem 4 membros.

Básico da struct

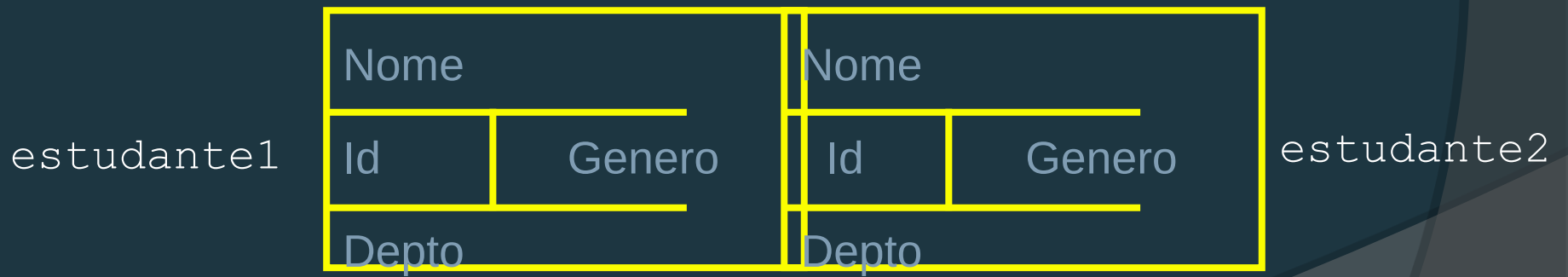
- **Declaração de uma variável do tipo struct:**

  `<struct-type> <identifier_list>;`

- **Exemplo:**

  `Estudante estudante1, estudante2;`

  `estudante1` **e** `Studante2` **são variáveis do tipo Estudante.**

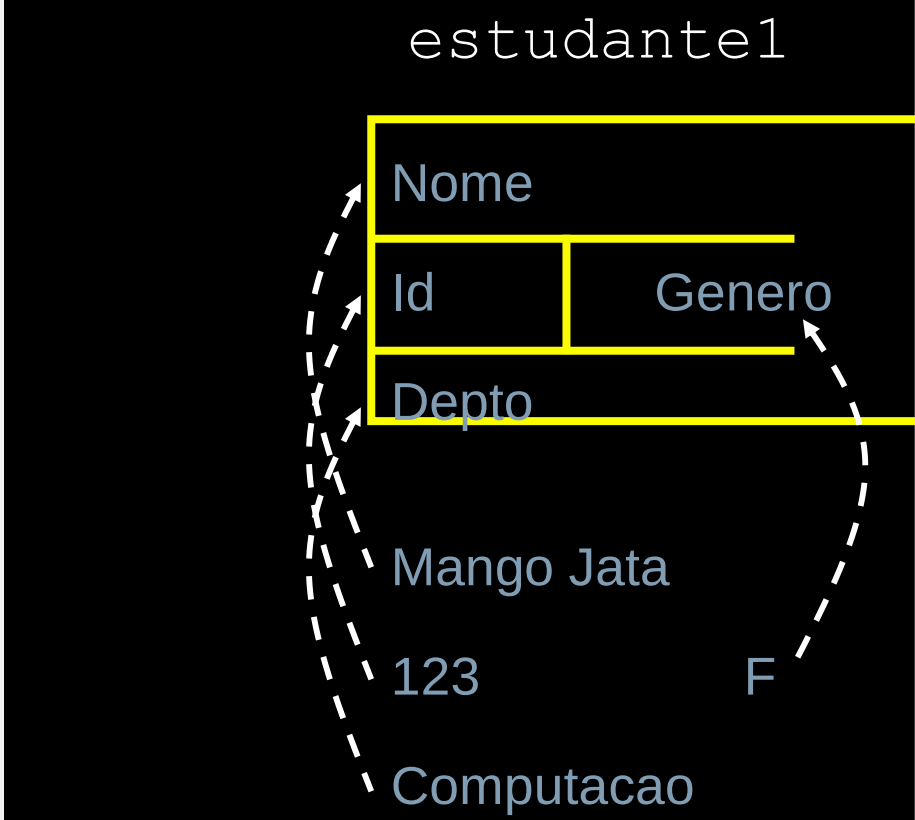


- O membro de uma variável tipo struct é acessado com operador ponto (.): `<variável_struct>.<nome_do_membro>;`

Exemplo 1:

```
#include <iostream>
#include <algorithm>
#include <cstdio>

using namespace std;

struct Estudante {
    string Nome;
    int Id;
    string Depto;
    char Genero;
};

int main() {
    Estudante estudante1, estudante2;
    estudante1.Nome = "Mango Jata";
    estudante1.Id = 123;
    estudante1.Depto = "Computacao";
    estudante1.Genero = 'F';
    cout << "O estudante eh ";
    if (estudante1.Genero=='M') {
      cout << "Sr. " << estudante1.Nome << endl;
    } else {
      cout << "Sra. " << estudante1.Nome << endl;
    }
    return(0);
}
```

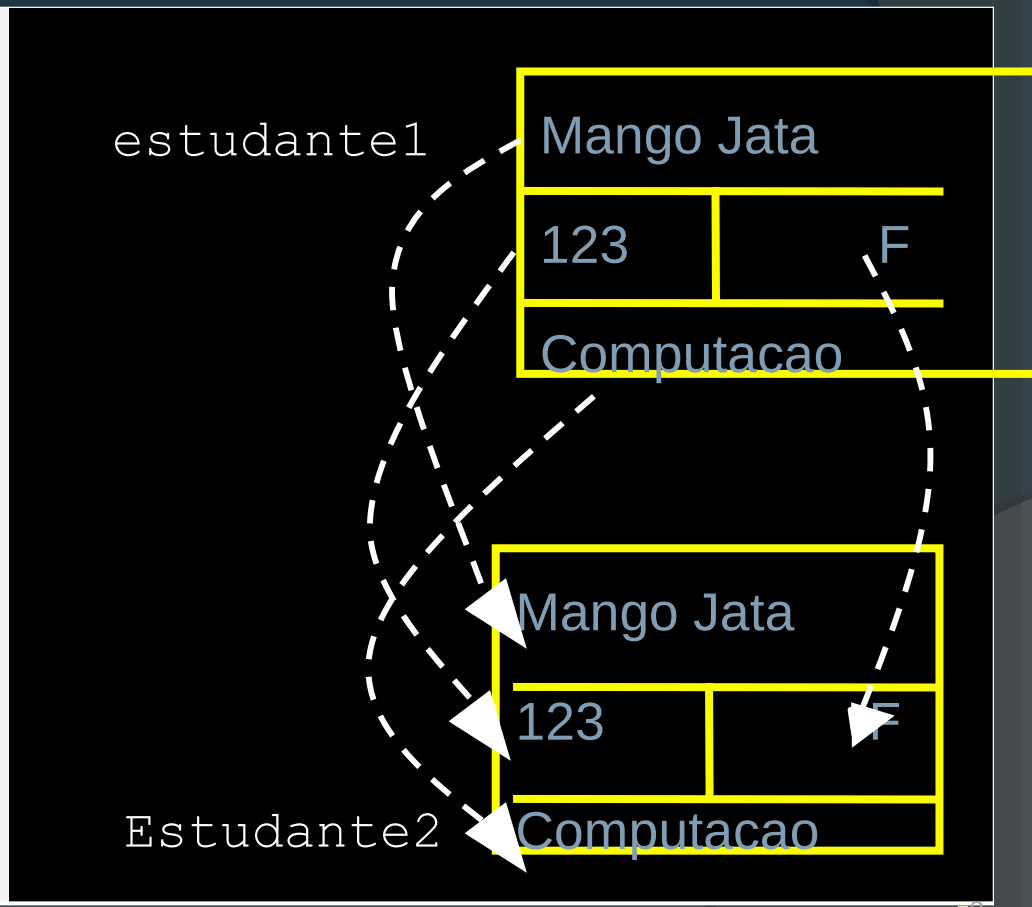
```
O estudante eh Sra. Mango Jata
```



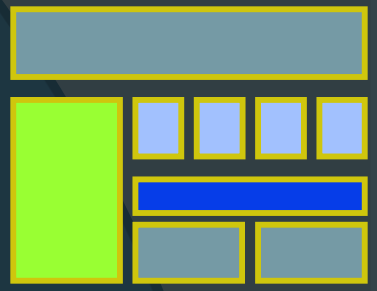
Exemplo 2: atribuição struct-to-struct

- **Os valores contido em uma variável do tipo estrutura podem ser atribuídos para outra variável do mesmo tipo. Exemplo:**

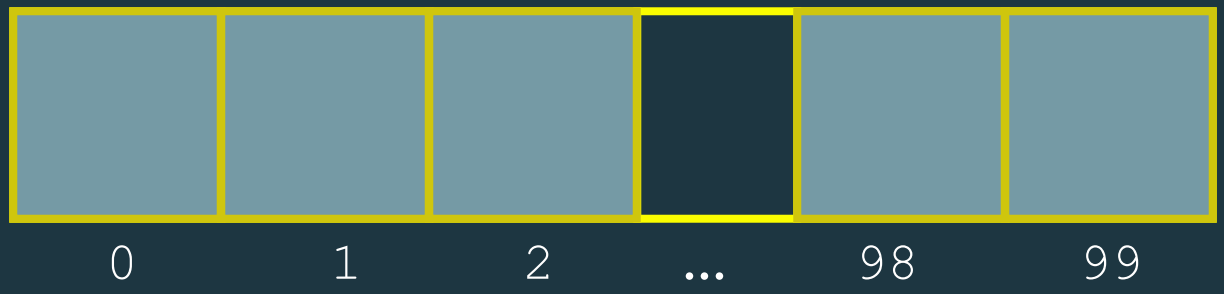
```
...

int main() {
    ...
    estudante2 = estudante1

    return(0);
}
```



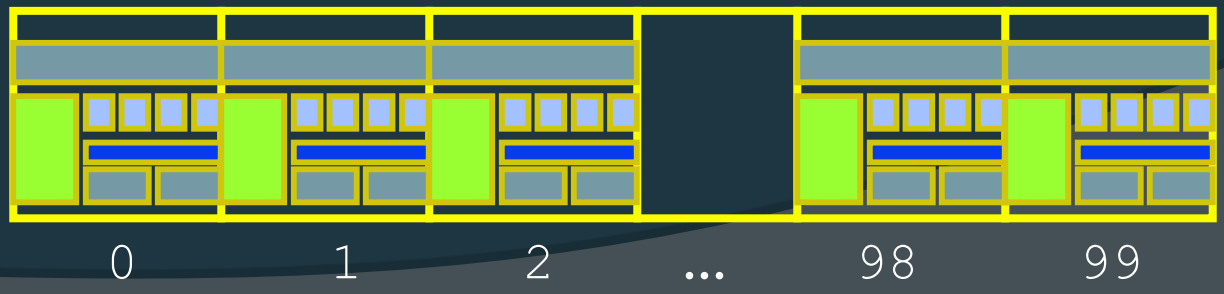
Arrays de estruturas



- **Um array comum: Um tipo de dado**



- **Array de structs: Múltiplos tipos de dados em cada elemento do array.**



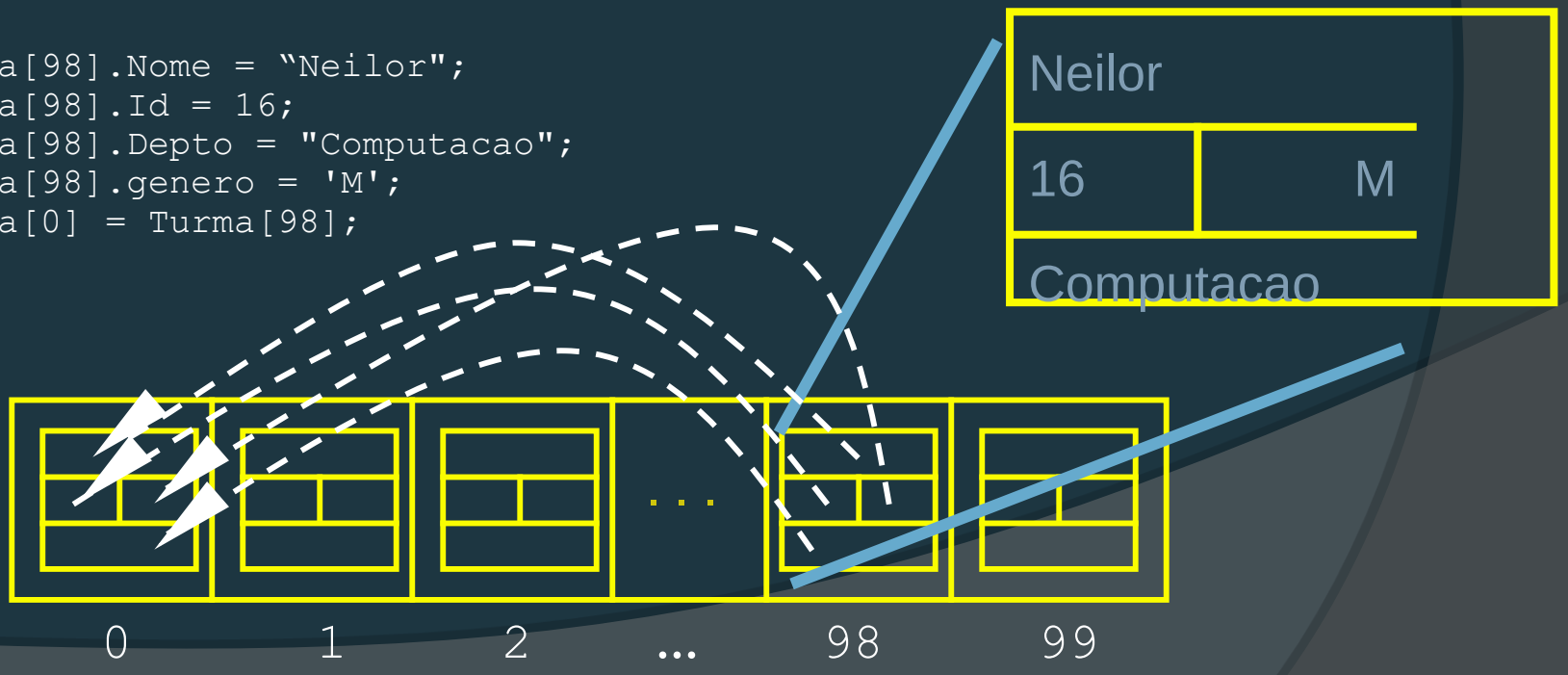
Arrays de estructuras

- **Muitas vezes utilizamos arrays de estruturas.**

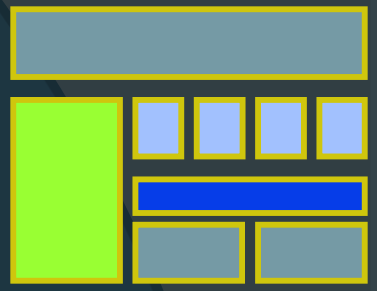
- **Exemplo:**

```
Estudante Turma[100];
```

```
Turma[98].Nome = "Neilor";
Turma[98].Id = 16;
Turma[98].Depto = "Computacao";
Turma[98].genero = 'M';
Turma[0] = Turma[98];
```



Exercício 1



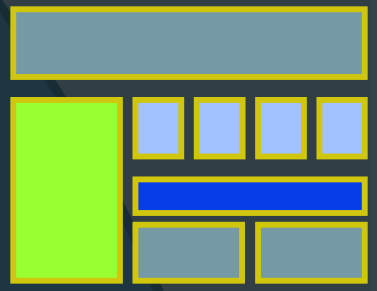
- **Leia os dados de 5 estudantes e apresente na tela da seguinte forma:**

```
Mango Jata – 123 – F – Computacao
Pedro Bo – 101 – M – Matematica
...
```

Exercício 2

- **Apresente as informações destes 5 estudantes ordenadas por ordem crescente do nome do estudante**

Exercício 3



- **Leia 6 Ids de um problema e seus respectivos tempos. Ex:**

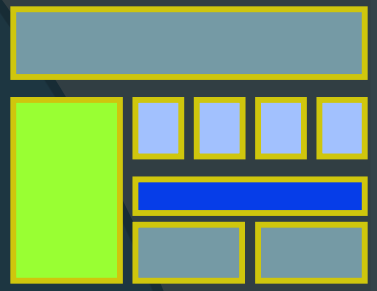
  URI 1024

  0.008

- **Em um mesmo programa, ordene por:**
  a) **Ordem alfabética ascendente de Id**
  b) **Ordem ascendente de tempo**
  c) **Ordem de tempo e para tempos iguais, ordem ascendente de Id.**

URI 1491
1.008
URI 1391
0.004
URI 1411
1.008
URI 1212
0.004
URI 1149
0.004
URI 1210
0.208

Exercício 4



- **Considerando a seguinte estrutura:**

```
typedef struct{
    int dia, mes, ano;
    string nome;
} Aniver;
```

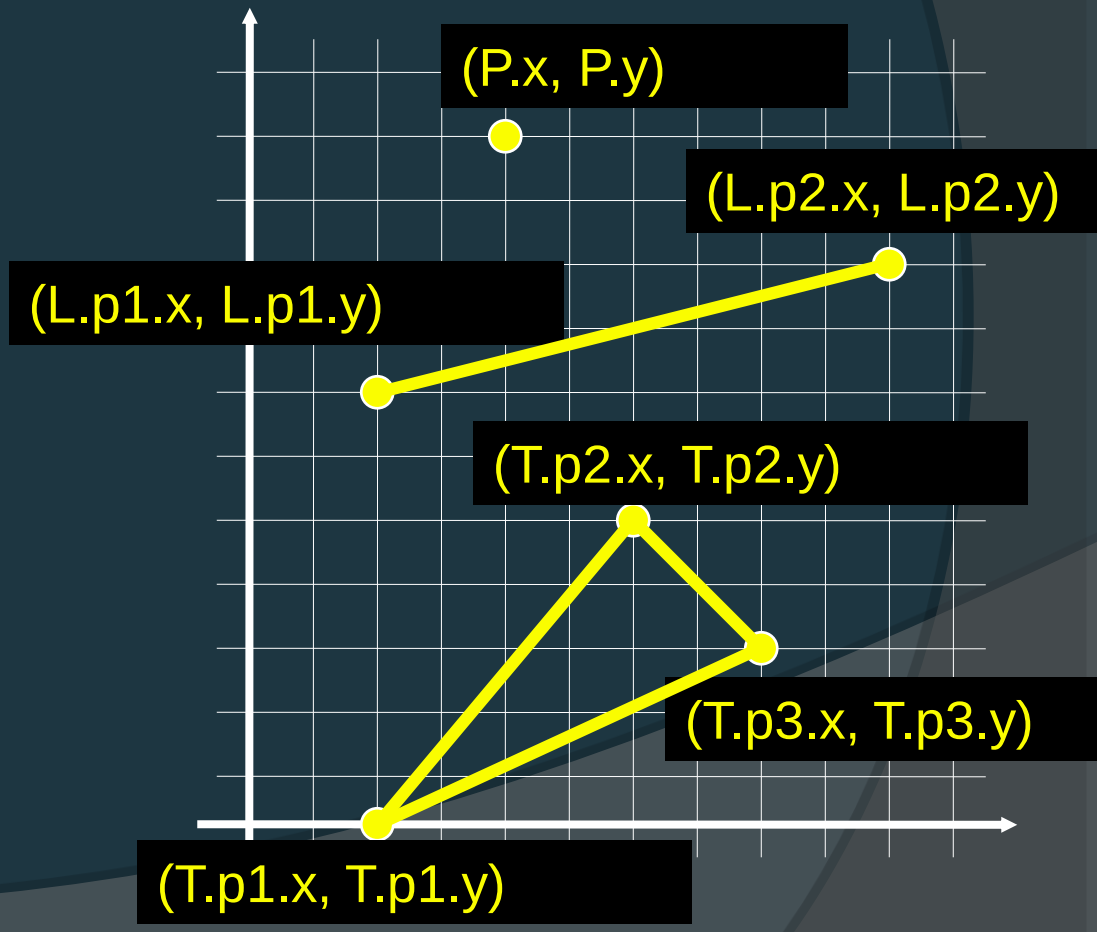
```
Saída após ordenação:
4 1 1980 - Antonio Prado
4 1 1980 - Juca Bala
4 1 1980 - Pedro Silva
10 1 1980 - Maria Antonia Silva
4 2 1980 - Cesar Torres
1 1 1991 - Andrea Souza
4 4 1992 - Cesar Bo
5 4 1992 - Pedro Bo
7 12 1992 - Joao Bo
2 1 1993 - Aristoteles Bo
```

- **Leia 10 nomes com as datas de aniversário. Em seguida ordene por:**
  - **Ordem ascendente de data de aniversário**
  - **Para datas iguais, ordene por ordem ascendente de nome**

Ex. 3-5: estruturas aninhadas

- **Podemos criar estruturas dentro de estruturas.**
- **Exemplos:**

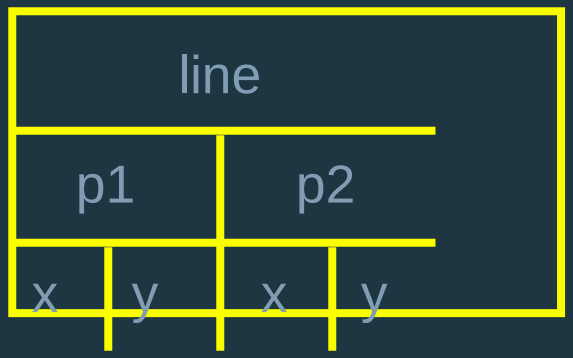
```
struct point{
    double x, y;
};
point P;

struct line{
    point p1, p2;
};
line L;

struct triangle{
    point p1, p2, p3;
};
triangle T;
```

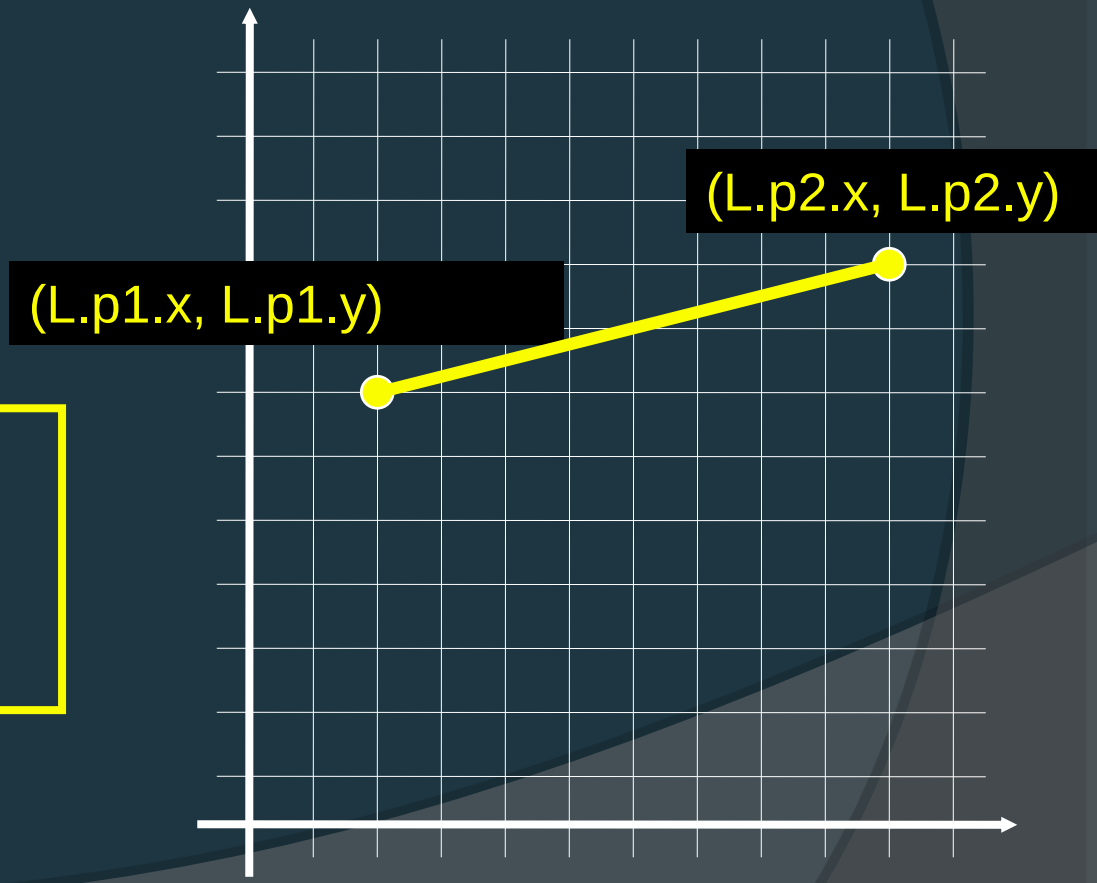


Ex. 3-5: Estruturas Aninhadas

- **Nós podemos criar estruturas dentro de estruturas:**

```
struct line{
    point p1, p2;
};
line L;
```

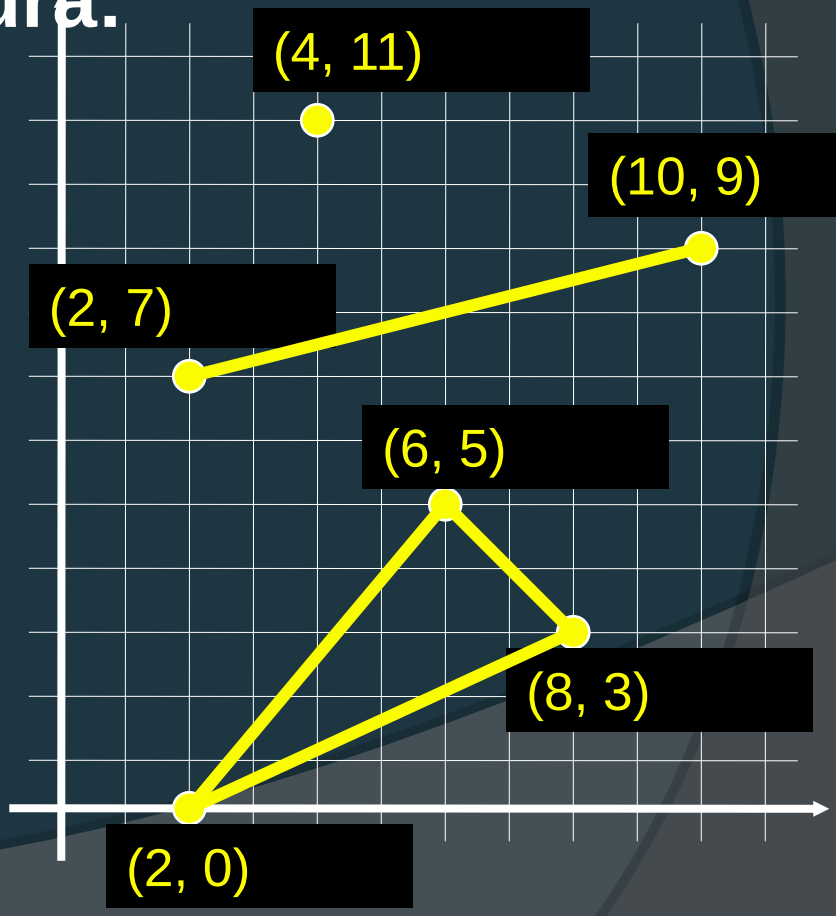




Ex. 3-5: Estruturas Aninhadas

- **Atribuindo valores para as variáveis `P`, `L`, and `T` usando a figura:**

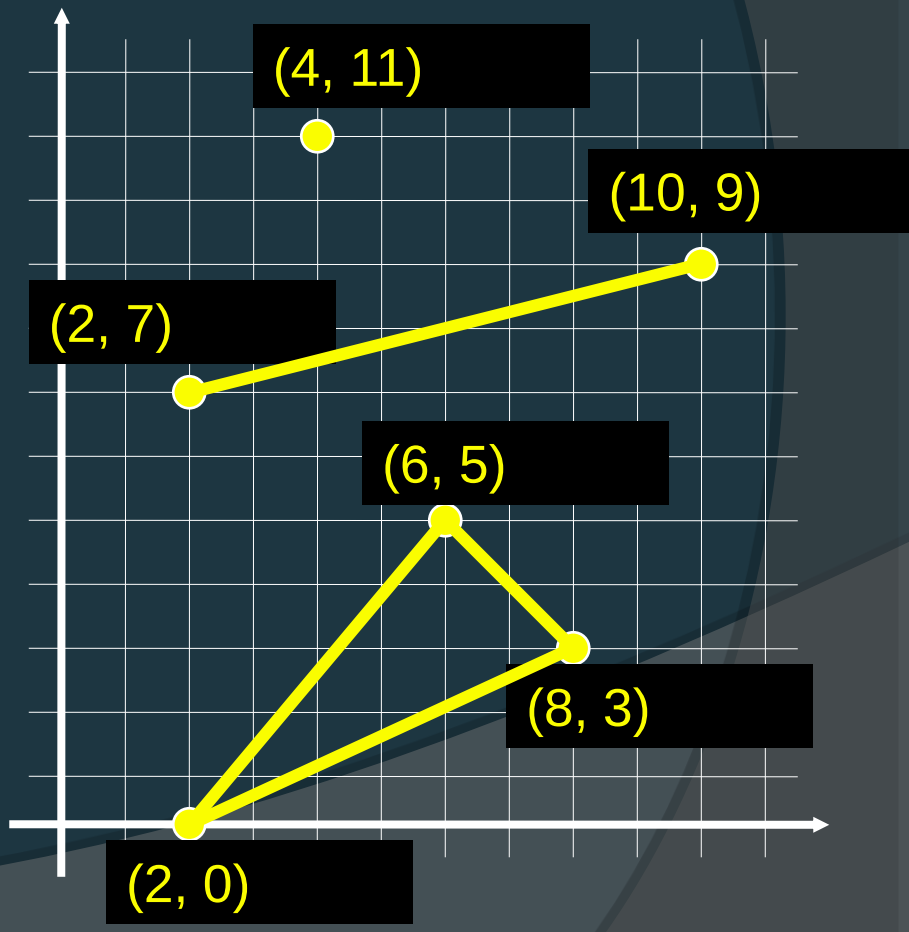
```
point P;
line L;
triangle T;
```



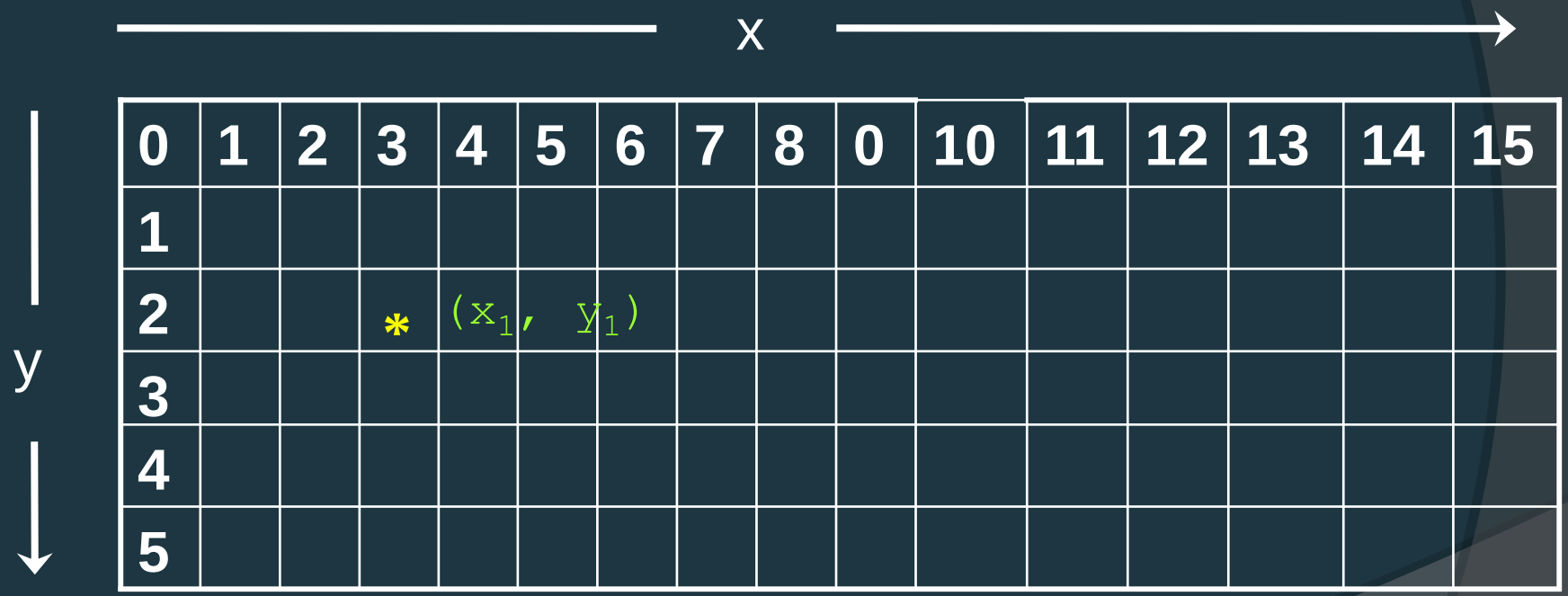
- **Ex. 3: desenhando um ponto**
- **Ex. 4: Desenhando uma linha**
- **Ex. 5: Desenhando um triângulo**

Ex. 3-5: Estruturas Aninhadas

```
point P;
line L;
triangle T;
P.x = 4;
P.y = 11;
L.p1.x = 2;
L.p1.y = 7;
L.p2.x = 10;
L.p2.y = 9;
T.p1.x = 2;
T.p1.y = 0;
T.p2.x = 6;
T.p2.y = 5;
T.p3.x = 8;
T.p3.y = 3;
```



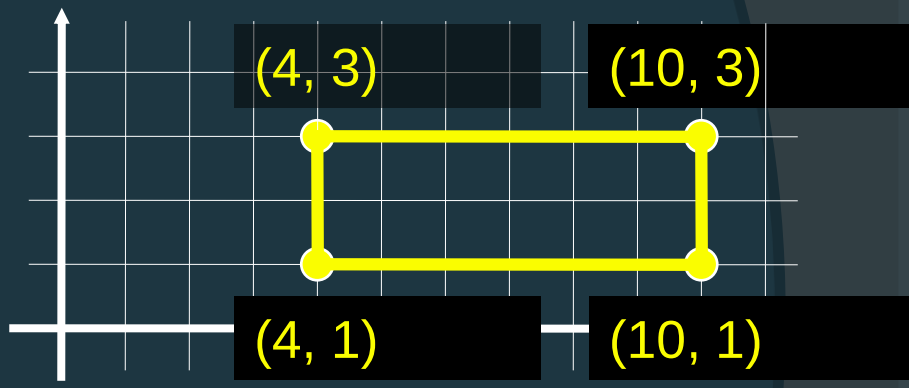
Ex. 3: Desenhando um ponto



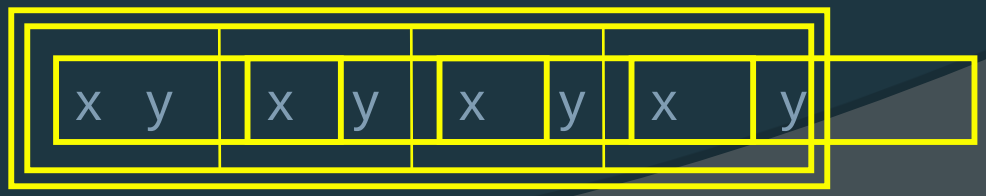
Arrays inside structures

- **We can use arrays inside structures.**
- **Example:**

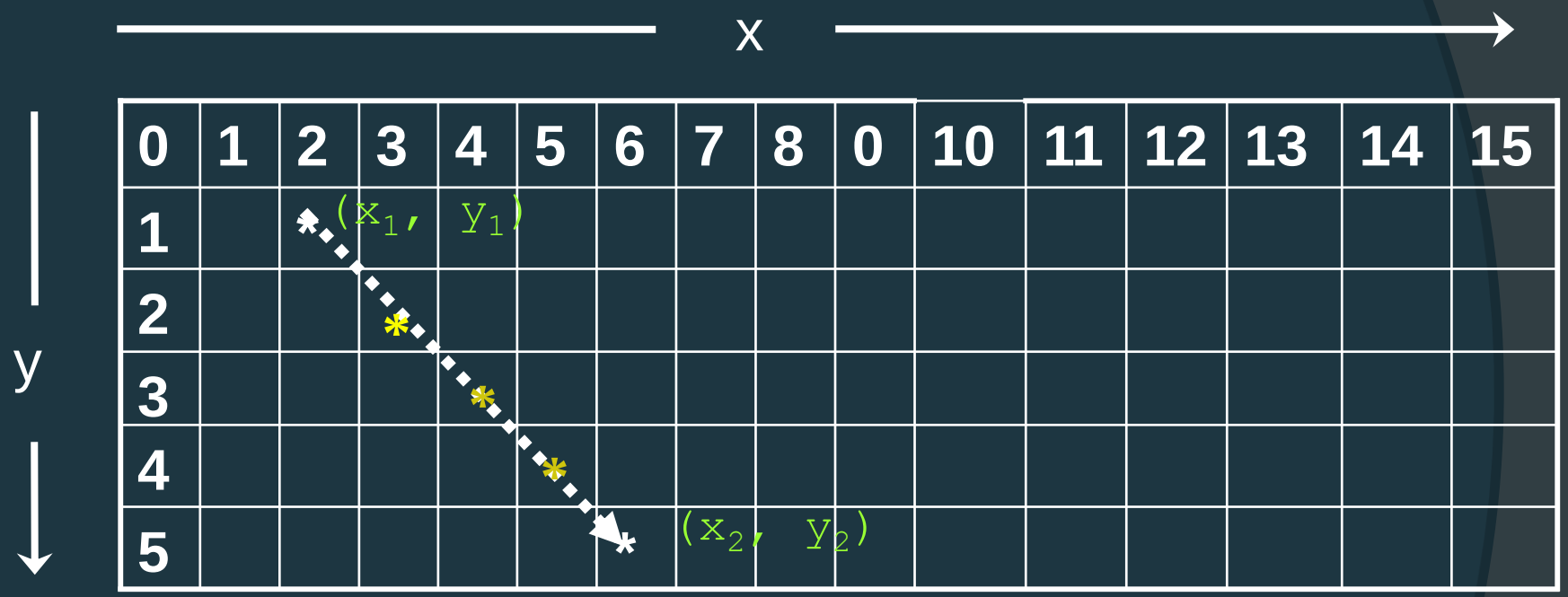
```
struct square{
    point vertex[4];
};
square Sq;
```



- **Assign values to `Sq` using the given square**



Ex. 4: Trajetória entre 2 pontos



- Rodar código:

  Struct_2.cpp