Programação Imperativa e Funcional

Dados Estruturados

erico.souza.teixeira
est@cesar.school

Cartas do Baralho

Estruturas

```
struct member{
    int int_member;
    double double_member;
    char string_member[25];
};

struct member var;
sizeof(struct member)
```

Estruturas

```
struct mystruct{
    int int_member;
    double double_member;
    char string_member[25];
} struct_var1,struct_var2,struct_var3;

struct_var1.int_member = 10;
struct_var2 = struct_var1;
struct_var3 = {10, 1.1, ”struct”}
```

Estruturas

```
struct {
    int int_member;
    double double_member;
    char string_member[25];
} struct_var;
```

Estruturas aninhadas

```
struct addr {
    char rua[40];
    int numero;
};
struct escola {
    char nome[40];
    struct addr endereco;
}school;
school.endereco.numero = 181;
```

Estruturas

```
typedef struct {
    int int_member;
    double double_member;
    char string_member[25];
} Mystruct;

Mystruct struct_var1, struct_var2;
```

Uniões

- **Posição de memória compartilhada por duas ou mais variáveis**

```
union tipo_u{
   int i;
   double d;
};
union tipo_u u;
u.i = 10;
```

Enumeradores

- Associação entre identificadores e inteiros

```
enum weather {
    sunny,
    windy,
    cloudy,
    rain,
};
enum weather weather_outside = rain;
printf("%d", weather_outside);
```

Enumeradores

- Associação entre identificadores e inteiros

```
enum weather {
    sunny,
    windy,
    cloudy=4,
    rain,
};
enum weather weather_outside = rain;
printf("%d", weather_outside);
```

Programação Imperativa e Funcional

Funções

erico.souza.teixeira
est@cesar.school

Fatorial

Funções, ou ...

- Sub-rotina
- Subprograma
- Procedimento
- Método

```
tipo nome(tipo_1 arg_1,
          tipo_2 arg_2,
          ...) {

  /* code */
}
```

Recursão

```
tipo nome(tipo_1 arg_1,
          tipo_2 arg_2,
          ...) {

  /* code */
  nome(arg1, arg2, ...);
  /* code */
}
```

Protótipos

```
tipo nome(tipo_1,
          tipo_2,
          ...);
```

```c
#include <stdio.h>
int abs(int);
int main() {
 int x, y;
 printf(”Digite as coordenadas x e y de um ponto:");
 scanf("%d,%d", &x, &y);
 printf(”A distância p/ o eixo x é %d.\n”,
         abs(x));
 printf(”A distância p/ o eixo y é %d.\n ”,
         abs(y));
 return 0;
}
int abs(int a) {
  if (a>=0) return a; else return -a;
}
```

Média de números inteiros

Número variado de argumentos

```
#include <stdarg.h>
float media (int n_args, ...) {
 va_list argumentos;
 va_start (argumentos, n_args);
 int contador = 0;
 int soma = 0;
 while (contador < n_args) {
  int numero = va_arg (argumentos, int);
  soma += numero;
  contador += 1;
 }
 va_end (argumento);
 float med = (float)(soma) / (float)(contador);
 return med;
}
```