

Fundação Hermínio Ometto

Bacharelado em Sistemas de Informação

SIF009 - Linguagem de Programação I

Prof. Dr. Sérgio Luis Antonello

Aula 08

22/09/2020

Plano de ensino

1. Unidade I - Programação estruturada e Linguagem C (objetivos b, c e d).
 - 1.1. Conceitos de programação estruturada.
 - 1.2. Estrutura de um programa de computador.
 - 1.3. Códigos fonte, objeto e executável.
 - 1.4. Biblioteca de códigos.
 - 1.5. Compiladores e Interpretadores.
 - 1.6. Processos de compilação e link edição.
 - 1.7. Identificação dos tipos de erros e alertas (léxicos, sintáticos e semânticos).
 - 1.8. Depuração de código.
 - 1.9. Palavras reservadas.
 - 1.10. Tipos de dados.
 - 1.11. Constantes. Variáveis simples e estruturadas. Escopo de variáveis.
 - 1.12. Operadores e precedência.
 - 1.13. Expressões aritméticas, lógicas e relacionais.
 - 1.14. Comandos.
 - 1.15. Ambientes de desenvolvimento e programação.
2. Unidade II - Estruturas de controle (sequência, decisão e repetição), registro e arquivo (objetivos a, c, d).
 - 2.1. Comandos if, for, while e switch.
 - 2.2. Blocos de comandos e aninhamento.
 - 2.3. Definição de tipos.
 - 2.4. Registro.
 - 2.5. Arquivo: leitura e gravação de dados em disco.
3. Unidade III - Ponteiros e Funções (objetivos a, c, d, e).
 - 3.1. Ponteiros.
 - 3.2. Funções.
 - 3.3. Passagem de parâmetro por valor.
 - 3.4. Passagem de parâmetro por referência.
4. Unidade IV- Strings e Variáveis indexadas (objetivos a, c, d).
 - 4.1. Manipulação de strings.
 - 4.2. Manipulação de caracteres.
 - 4.3. Declaração e manipulação de vetores.
 - 4.4. Declaração e manipulação de matrizes.

Plano de ensino

| Data | Atividade |
|-------|-----------|
| 04/08 | Aula 01 |
| 11/08 | Aula 02 |
| 18/08 | Aula 03 |
| 25/08 | Aula 04 |
| 01/09 | Aula 05 |
| 08/09 | Aula 06 |
| 15/09 | Prova 1 |
| 22/09 | Aula 08 |
| 29/09 | Aula 09 |
| 06/10 | Aula 10 |

| Data | Atividade |
|-------|-----------------------------|
| 13/10 | Aula 11 |
| 20/10 | Maratona FHO de Programação |
| 27/10 | Aula 13 |
| 03/11 | Aula 14 |
| 10/11 | Aula 15 |
| 17/11 | Prova 2 Entrega Trabalho |
| 24/11 | Prova SUB |
| 01/12 | Semana Científica |
| 08/12 | Aula 19 |
| 15/12 | Aula 20 |

Sumário da aula

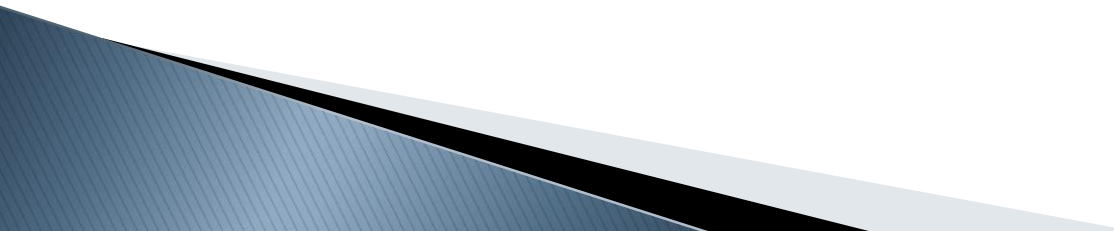
Primeiro momento (revisão)

- ✓ Devolutiva da P1

Segundo momento (conteúdo)

- ✓ Registro
- ✓ Definição de tipo
- ✓ Leitura e gravação em disco

Terceiro momento (síntese)

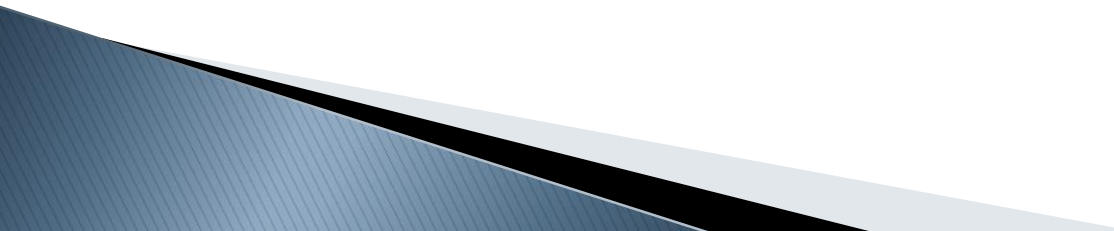
- ✓ Retome pontos importantes da aula
- 

1. Primeiro momento: revisão

- ▶ Devolutiva da prova1



2. Segundo momento: motivação

- ❑ O que é o Schoolnet?
 - ❑ Facebook
 - ❑ Internet banking
 - ❑ Uma conta corrente
- 

2. Segundo momento

- ▶ Registro
- ▶ Definição de tipo
- ▶ Leitura e gravação em disco



3. Registro com uso de struct

Um registro em C ou estrutura é um tipo de dado estruturado heterogêneo.

É uma coleção de variáveis referenciadas sob um único nome.

Mantém juntas informações logicamente relacionadas.

Possibilita tratar um grupo de valores como uma única variável.

Sintaxe:

```
struct nome_da_estrutura {  
    tipo_1 nome_va1;  
    tipo_2 nome_2;  
    ...  
    tipo_n nome_n;  
} nome_das_variáveis_da_estrutura;
```


3. Registro com uso de struct

Exemplos:

```
struct triang {  
    float base;  
    float altura;  
};
```



“base” é uma parte da estrutura “triang”.

Cada parte da estrutura é conhecida como campo ou atributo.

```
struct retang {  
    float lado1;  
    float lado2;  
    float area;  
};
```

3. Registro com uso de struct

Exemplo:

```
struct endereco {  
    char rua [50];  
    int numero;  
    char bairro [20];  
    char cidade [30];  
    char UF [2];  
    char CEP[9];  
};
```

3. Registro com uso de struct

Exemplo:

```
struct endereco {  
    char rua [50];  
    int numero;  
    char bairro [20];  
    char cidade [30];  
    char UF [2];  
    char CEP[9];  
};
```

```
struct ficha_pessoal {  
    char nome [50];  
    long int telefone;  
    struct endereco end;  
};
```

Variável “end” armazena dentro dela conteúdos da estrutura endereco, ou seja, rua, numero, bairro, cidade, UF e CEP.



3. Registro com uso de struct

```
struct {
```

```
int a;  
char b;  
float c;  
int v[5];
```

```
} x;
```

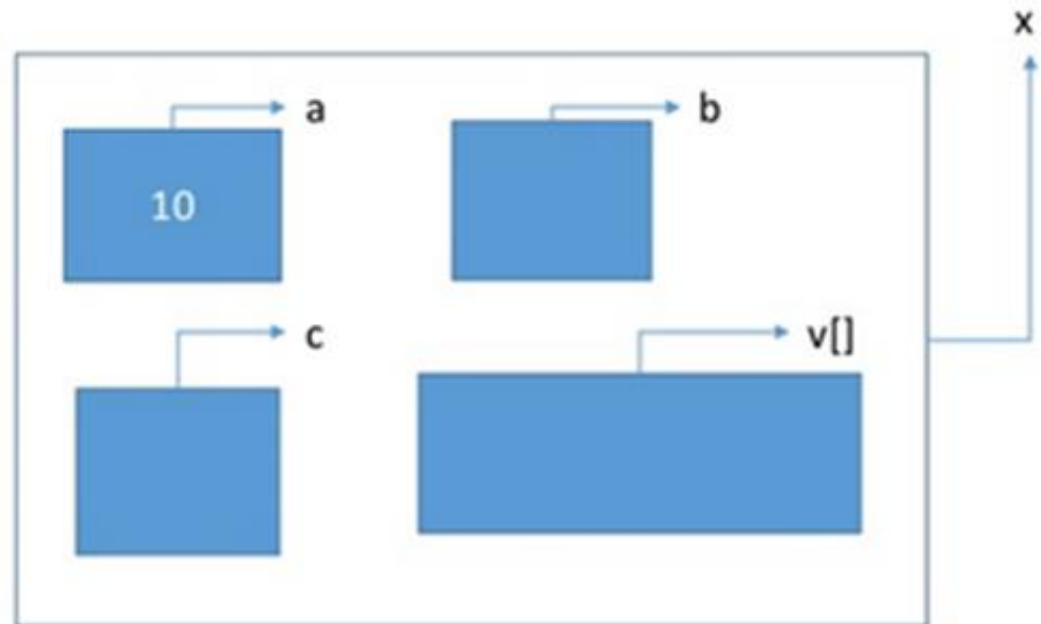
x é a variável.

a, b, c e v são partes de x.

Se for executado o comando

```
x.a = 10;
```

a parte da estrutura camada de “a” recebe o conteúdo 10.




4. Typedef: definição de tipo

Além dos tipos oferecidos pela linguagem C, o uso do **typedef** possibilita que o programador defina tipos específicos dentro do seu programa.


```
typedef struct _p{  
    int x;  
    int y;  
} ponto;
```

“ponto” é o novo tipo que pode ser usado pelo programa. Este tipo é uma struct que contém dois atributos, “x” e “y” que são do tipo inteiro.



```
int main() {  
    ...  
    ponto p1, p2;  
    ...  
}
```


“p1” e “p2” são variáveis que estão sendo declaradas como tipo “ponto”.



4. Typedef: definição de tipo

Além dos tipos oferecidos pela linguagem C, o uso do **typedef** possibilita que o programador defina tipos específicos dentro do seu programa.


```
typedef struct _p{  
    int x;  
    int y;  
} ponto;
```



“ponto” é o novo tipo que pode ser usado pelo programa. Este tipo é uma struct que contém dois atributos, “x” e “y” que são do tipo inteiro.

```
int main() {  
    . . .  
    ponto p1, p2, p3;
```

Declaração de três
variáveis do tipo
“ponto”.



```
    p1.x = 10;  
    p1.y = 20;  
    p2.x = p1.x + 5;  
    p2.y = p2.y + 5;  
    p3 = p2;  
  
}
```

5. I/O em arquivo texto

Usando comandos específicos da linguagem C, veremos como ler ou gravar dados em arquivo texto no disco.

- Streams são usados para ler e escrever na linguagem C.
- Um stream pode ser um dispositivo padrão do sistema, como o vídeo, ou um arquivo específico.
- Quando um arquivo é aberto através da função “fopen” ocorre uma associação do mesmo com um stream.
- Ao fazer uso do “fclose”, esta associação é quebrada e força uma transferência dos dados da área de buffer para o arquivo.

5. I/O em arquivo texto

Dentre outros comandos (funções) veremos:

- `fopen()` abre um arquivo.
- `fclose()` fecha um arquivo.
- `fprintf()` saída de dados.
- `fscanf()` leitura de dados.
- `fgets()` ler uma string.
- `fputs()` gravar uma string.
- `ferror()` retorna verdadeiro se ocorreu erro.
- `feof()` retorna verdadeiro se encontrou fim de arquivo.



5. I/O em arquivo texto

A função `fopen()` utiliza um parâmetro que caracteriza a forma de abertura de um arquivo. Dentre outros, veremos:

- “r” abre um arquivo para leitura.
- “w” cria um arquivo para gravação.
- “a” acrescenta dados a um arquivo já existente.
- “r+” abre um arquivo para leitura/ gravação.
- “w+” cria um arquivo para leitura/ gravação.

5. I/O em arquivo texto

Exemplo:

```
int main () {
    FILE *fp;
    char nome[40];

    fp = fopen ("c:\\lpi\\nome.txt", "w");
    if ((fp == NULL) {
        printf ("o arquivo nao pode ser aberto\n");
        exit (1);
    }
    gets(nome);           // entrada a partir do teclado
    fputs(nome, fp);      // saída no disco
    fclose (fp);
}
```

5. I/O em arquivo texto

Exemplo 2: ler dois números em um arquivo e gravar o resultado da multiplicação entre eles em outro arquivo.

```
int main () {
    FILE *in, *out;
    ...
    in = fopen ("c:\\lpi\\entrada.txt", "r");
    out = fopen ("c:\\lpi\\saida.txt", "w");

    if ((in == NULL) || (out == NULL)) {
        printf("Problema ao abrir um dos arquivos\n");
        exit (1);
    }

    fscanf(in, "%d %d" , &a, &b);
    fprintf (out, "%d" , a*b);
    fclose(in);
    fclose(out);
}
```

6. Exercícios

**Vamos
Programar!**



6. Exercícios

- 1) Baseado nos exemplos de dados de entrada dispostos no arquivo `retang.in` (valores inteiros), usando os conceitos de registro, escreva um programa que calcule o perímetro e a área de retângulos.

Cada linha deste arquivo contém os valores de entrada de dados para os lados A e B de um retângulo.

A entrada de dados termina quando pelo menos um dos valores dos lados A e B for igual a zero.

6. Exercícios

2) Desenvolver um programa em linguagem C, que possibilite calcular a área de vários triângulos, cujas bases e alturas são obtidas através do arquivo triang.in.

Cada linha do arquivo contém dois dados (float), a base e a altura de um triângulo.

A entrada de dados encerra-se quando o valor da base ou da altura for igual a zero.

OBS: Apresentar a saída de dados em um arquivo texto.



6. Exercícios

3) Para o problema URI 2221

<https://www.urionlinejudge.com.br/judge/pt/problems/view/2221>

Usar dectype e struct para declarar um tipo de dados com os atributos:

- ✓ ValorGolpe
- ✓ Ataque
- ✓ Defesa
- ✓ Level

Usar o ambiente URI online Judge para validar o código desenvolvido.

7. Terceiro momento: síntese

- O typedef permite definir um tipo de dados a ser usado no programa.
- O struct possibilita especificar um dado estruturado (composto por mais de uma variável).
- Ler e escrever dados em um arquivo texto na linguagem C é feito por meio de stream. Os comandos básicos para isso são `fopen()`, `fclose()`, `fscanf()` e `fprintf()`.