Linguagem de Programação

Estruturas e Definição de Tipo

(*struct / typedef*)

Gil Eduardo de Andrade

Estruturas (*struct*)

Introdução

- Uma estrutura, em C, é uma coleção de variáveis referenciada por apenas um nome;
- As estruturas fornecem uma maneira conveniente para se agrupar informações que possuem algum tipo de relação entre si;

Estruturas (*struct*)

Introdução

- As informações agrupadas nas estruturas são representadas (armazenadas) em variáveis;
- Tais variáveis são denominadas membros da estrutura;

Estruturas (*struct*)

Exemplo:

Declara uma nova *estrutura* ou modelo de dados, especificando um identificador (*"aluno"*) para ela.

```
struct aluno {

        char nome[50];
        char curso[30];
        int turma;
};
```

A nova estrutura *"aluno"* possui como membros: *nome, curso e turma.*

Estruturas (*struct*)

Criando Estruturas

- Ao definirmos uma estrutura temos um modelo do qual é possível gerar variáveis;
- Ao declararmos uma variável do tipo estrutura, o compilador C aloca memória o suficiente para acomodar todos os seus membros;

Estruturas (*struct*)

Criando Estruturas

- Considerando que as estruturas são definidas para que seja possível criar um novo modelo de dados elas são declaradas globalmente;
- Isso permite que esse novo modelo de dados seja usado em todo o arquivo fonte C que está sendo codificado ;

Estruturas (*struct*)

Exemplo: criando variável estrutura

```c
#include <stdio.h>

struct aluno {

        char nome[50];
        char curso[30];
        int turma;
};

int main() {

        struct aluno sala[20];

        printf("\n");
        return 0;
}
```

Declara um novo modelo de dados (estrutura) denominado *"aluno"*, que possui como membros: *nome, curso e turma.*

Declara a variável *"sala"* do novo modelo de dados (estrutura) criado.

Estruturas (*struct*)

Manipulando os membros de uma estrutura

- Para que seja possível manipular (acessar) os membros de uma variável estrutura é utilizado o caractere reservado ponto ***"."***;
- Ou seja, considerando o exemplo anterior, para acessar o membro ***"turma"*** da variável estrutura ***"sala"*** teríamos: "***sala.turma";***

Estruturas (*struct*)

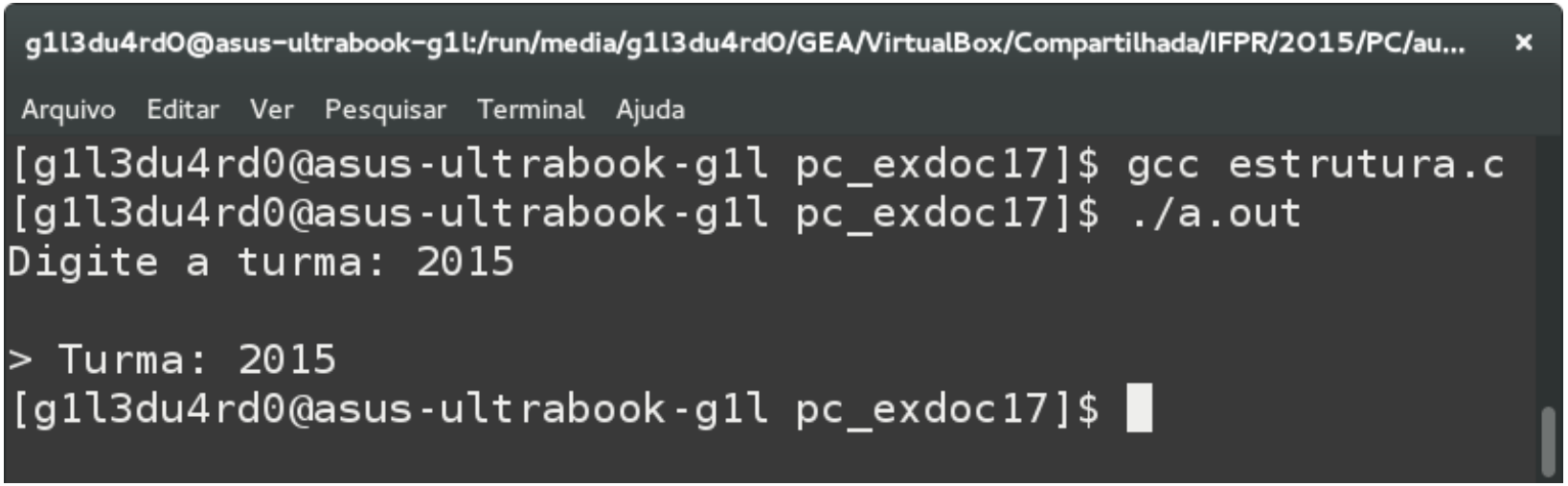
Exemplo: manipulando os membros de uma estrutura

```c
#include <stdio.h>

struct aluno {

	char nome[50];
	char curso[30];
	int turma;
};

int main() {

	struct aluno sala[20];

	printf("Digite a turma: ");
	__fpurge(stdin);
	scanf("%i", &sala[0].turma);

	printf("\n> Turma: %i", sala[0].turma);

	printf("\n");
	return 0;
}
```

Através do caractere reservado *"."* é possível acessar (manipular) o membro *"turma"* da variável estrutura *"sala"*.

Estruturas (*struct*)

Exemplo: manipulando os membros da estrutura

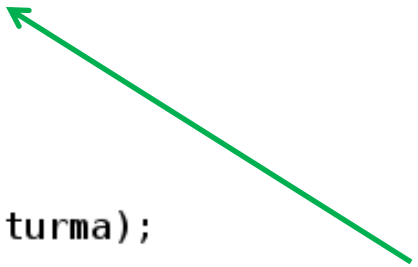


Resultado da execução.

Estruturas (*struct*)

Exemplo: alocação dinâmica de estruturas

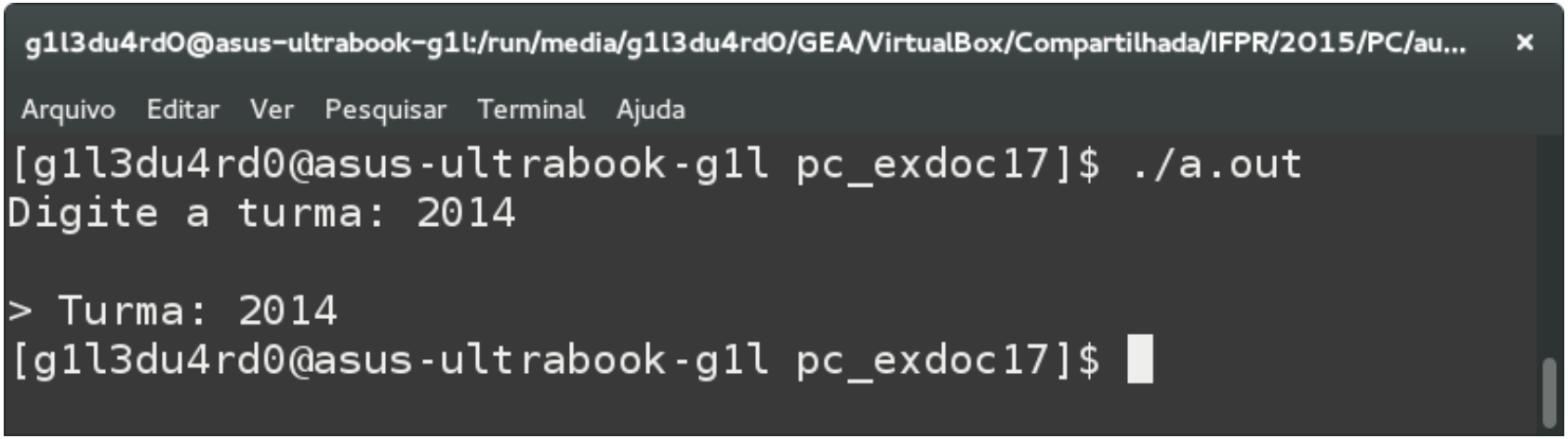
```
int main() {

      struct aluno *sala = malloc(20*sizeof(struct aluno));

      printf("Digite a turma: ");
      __fpurge(stdin);
      scanf("%i", &sala[0].turma);

      printf("\n> Turma: %i", sala[0].turma);

      printf("\n");
      return 0;
}
```



Utilizando o comando *"malloc"* é possível alocar memória para uma variável estrutura, assim como acontece com variáveis de outros tipos (*int, char*).

Estruturas (*struct*)

Exemplo: alocação dinâmica de estruturas



Resultado da execução.

Estruturas (*struct*)

Exemplo: passagem de estruturas para funções

```c
void recebe_estutura(struct aluno *sala);

int main() {

      struct aluno *sala = malloc(20*sizeof(struct aluno));

      printf("Digite a turma: ");
      __fpurge(stdin);
      scanf("%i", &sala[0].turma);

      recebe_estrutura(sala);

      printf("\n");
      return 0;
}

void recebe_estrutura(struct aluno *sala) {
      printf("\n> Turma: %i", sala[0].turma);
}
```

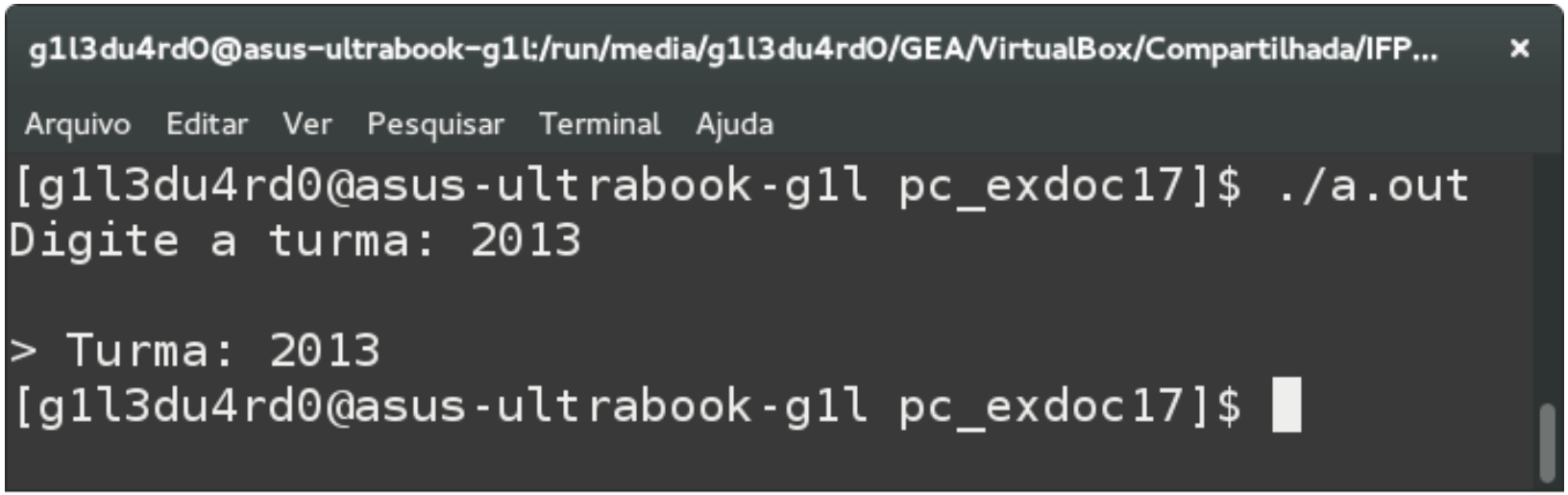
Declara função que recebe uma estrutura como parâmetro.

Invoca a função e passa a estrutura *"sala"* como parâmetro.

Implementação da função que recebe e acessa os dados da estrutura passada como parâmetro.

Estruturas (*struct*)

Exemplo: passagem de estruturas para funções



Resultado da execução.

Definição de Tipo (*typedef*)

Conceito

- As estruturas permitem que novos modelos de dados sejam criados;
- Tais modelos de podem ser definidos como um novo tipo, funcionando de forma análoga aos tipos nativos da linguagem C: ***int, char, float, double, etc***;

Definição de Tipo (*typedef*)

Typedef:

- Para que seja possível definir um novo tipo em C é utilizada a palavra reservada ***typedef***;
- O ***typedef*** deve ser utilizado juntamente com a declaração do novo modelo de dados (estrutura);

Definição de Tipo (*typedef*)

Exemplo: definindo o tipo aluno

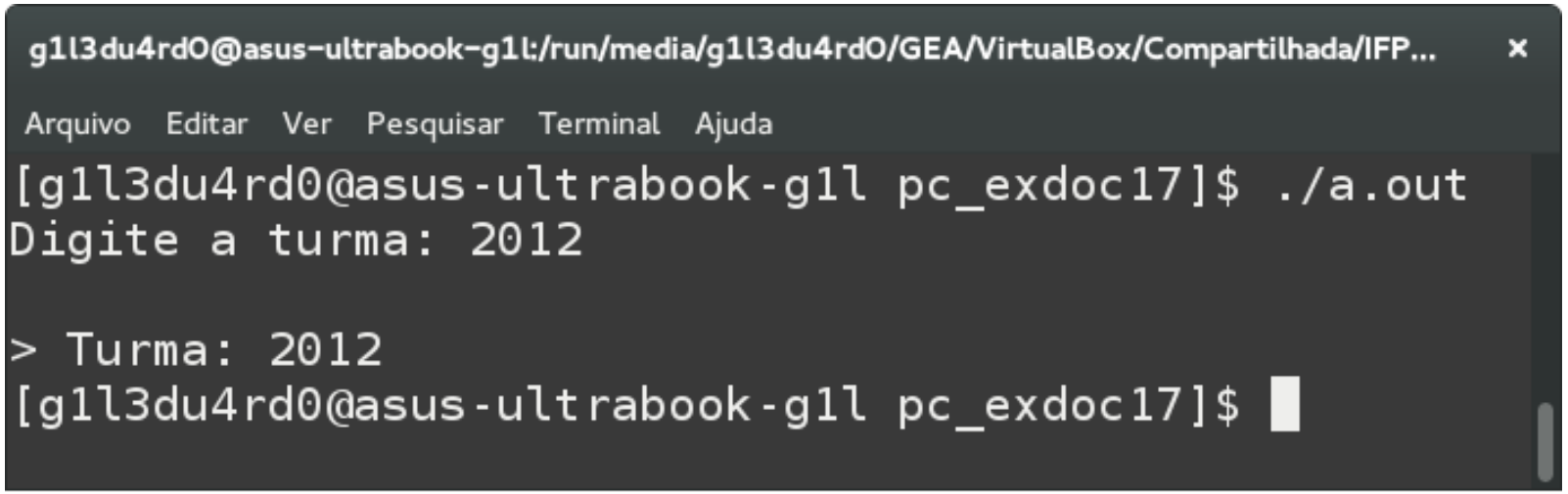
```c
#include <stdio.h>

typedef struct {

      char nome[50];
      char curso[30];
      int turma;
} aluno;

int main() {

      aluno sala[20];

      printf("Digite a turma: ");
      __fpurge(stdin);
      scanf("%i", &sala[0].turma);

      printf("\n> Turma: %i", sala[0].turma);

      printf("\n");
      return 0;
}
```

Diferentemente do exemplo anterior, a palavra reservada *"typedef"* é utilizada antes da palavra reservada *"struct"*. Já o nome do novo tipo definido é colocado ao final da declaração dos membros da estrutura.

Ao declararmos uma variável para o novo tipo definido (*aluno*) já não é mais necessário utilizar a palavra reservada *struct*. O procedimento torna-se similar a declaração de variáveis do tipo: *int, float, char,* etc.

Estruturas (*struct*)

Exemplo: definindo o tipo aluno



Resultado da execução.

Definição de Tipo (*typedef*)

Exemplo: definindo o tipo aluno - alternativo

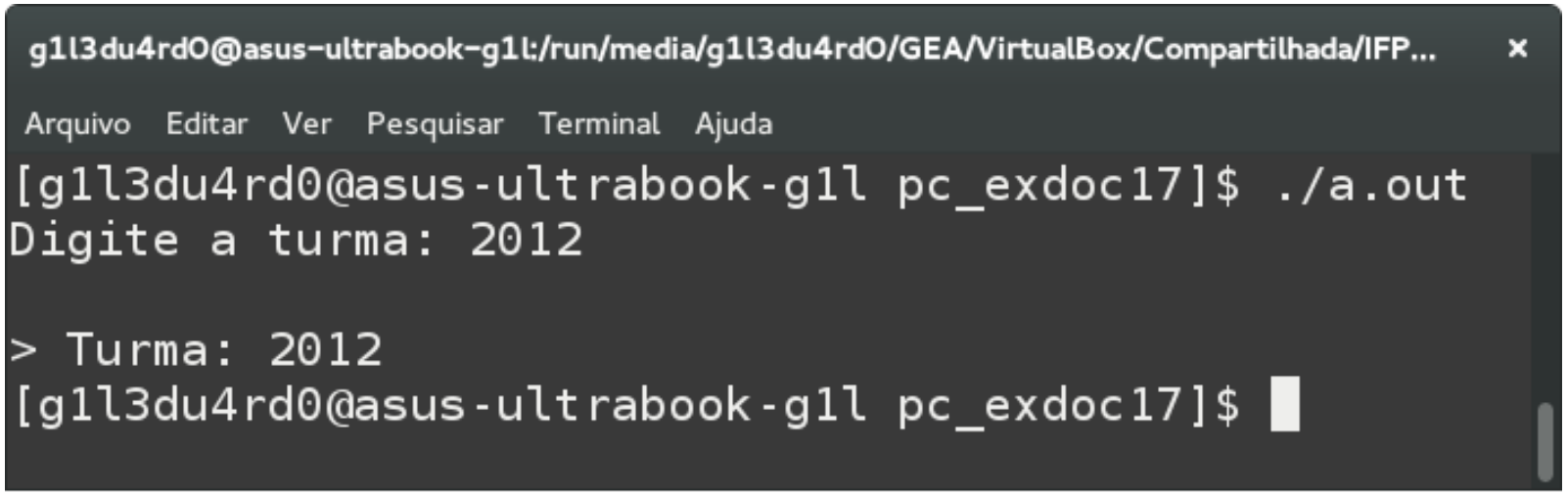
```c
#include <stdio.h>

struct dados {

        char nome[50];
        char curso[30];
        int turma;
};

typedef struct dados aluno;

int main() {

        aluno sala[20];

        printf("Digite a turma: ");
        __fpurge(stdin);
        scanf("%i", &sala[0].turma);

        printf("\n> Turma: %i", sala[0].turma);

        printf("\n");
        return 0;
}
```

Diferentemente do exemplo anterior, primeiramente é declarada a estrutura (*"struct"*) dados. Apenas após a declaração da estrutura é que a palavra reservada *"typedef"* é utilizada para criar um novo tipo de dado, denominado aluno.

Ao declararmos uma variável para o novo tipo definido (*aluno*) já não é mais necessário utilizar a palavra reservada *struct*. O procedimento torna-se similar a declaração de variáveis do tipo: *int, float, char,* etc.

Estruturas (*struct*)

Exemplo: definindo o tipo aluno - alternativo



Resultado da execução.

Estrutura e Definição de Tipo

Exemplos Utilizados no Documento

http://www.gileduardo.com.br/ifpr/pcii/downloads/pc_exdoc17.zip

Mais Exemplos sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/pcii/downloads/pc_ex17.zip

Exercícios sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/lp/downloads/lp_pratica17.pdf