UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Fundamentos de Programação

CP41F

Tipos básicos de dados.
Modificadores de tipos.

Aula 6

Prof. Daniel Cavalcanti Jeronymo

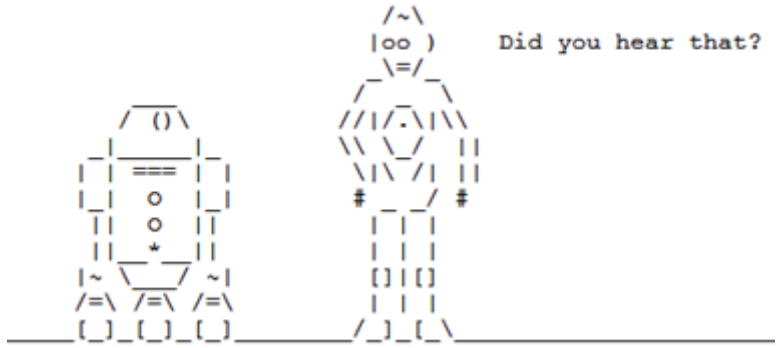
Universidade Tecnológica Federal do Paraná (UTFPR)
Engenharia de Computação – 1º Período

Plano de Aula

- Tipos básicos de dados
  - int, char, float, double, void
- Tipos definidos pelo usuário
  - struct, union, enum, typedef
- Tipos derivados
  - [ponteiros, vetores, funções]
- Modificadores de tipos
  - Tamanho, sinal, constante, volatilidade, armazenamento
  - [ponteiros, funções e interrupções]
- Literais
- limits.h

Tipos básicos de dados

- Caractere (char)
  - Tamanho: tipicamente, porém não restrito, a um byte (8 bits) – número de bits da arquitetura é encontrado na macro CHAR_BIT (limits.h)
  - Restrição: maior ou igual a 8 bits



  - Uso: representação de valores de caracteres utilizando alguma codificação (e.g. ASCII)
  - sizeof retorna o tamanho da unidade em chars

Tipos básicos de dados

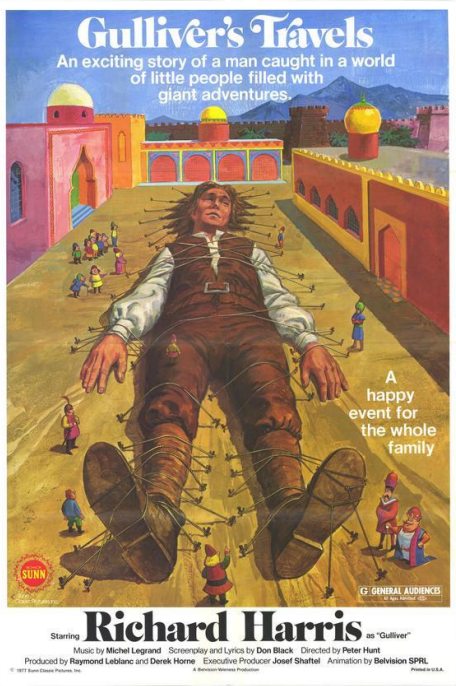
- sizeof
  - Não é uma função! é uma palavra chave
  - Retorna o tamanho, em chars, da variável ou do tipo
  - Uso:
    - sizeof(int);
    - sizeof int;
    - int x; sizeof(x);

Tipos básicos de dados

- Inteiro (int)
  - Tamanho: igual a unidade natural da arquitetura (word)
  - Restrição de tamanho: maior ou igual a 16 bits
  - Armazenamento mínimo: valores entre -32767 ~ 32767
  - Armazenamento depende da *endianness*

Tipos básicos de dados

- Inteiro (int)
  - Endianness – sequência de armazenamento de valores maiores que um byte (não se aplica a registradores!)



Origem do termo: Jonathan Swift e a sátira As Viagens de Gulliver, guerra centenária entre o reinado de Lilliput e o reinado de Blefuscu

Motivo: habitantes de Lilliput comiam ovos cozidos primeiro pela parte pequena (little endian) enquanto em Blefuscu os ovos eram iniciados pela parte grande (big endian)

Tipos básicos de dados

- Inteiro (int)
  - Endianness – Valor $90AB12CD_{16}$
  - big endian

byte mais significante

Address	Value
1000	90
1001	AB
1002	12
1003	CD

little endian

byte menos significante

Address	Value
1000	CD
1001	12
1002	AB
1003	90

Tipos básicos de dados

- Inteiro (int)
  - Endianness – Importância na transferência de informação entre arquiteturas

```
#include <stdio.h>
#include <string.h>

int main()
{
   int valor = 0x1A2B3C4D;
   int byte = 0;

   memcpy(&byte, ((char*)&valor) + 0, sizeof(char));

   printf("%X %X\n", valor, byte);

   return 0;
}
```

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Como representar números reais em binário?

  - Pense em decimal:
    - 14.75
    - 1*10^1 + 4*10^0 + 7*10^-1 + 5*10^-2

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Lembrete de aulas passadas: a operação de multiplicar por dois é equivalente, em binário, a deslocar bits para a esquerda (SHL)
  - 14.75 * 2^2 * 2^-2 (SHL)
  - = 59.0 * 2^-2
  - Deslocamos todo o conteúdo decimal para antes do ponto!

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Continuando, traduzimos o valor pra binário
  - Mantendo a notação de base para binário e ignorando para decimal (afim de simplificar)
  - = 59.0 * 2^-2
  - = 111011_2 * 2^-2

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Agora deslocando os bits para a direita (SHR),dividindo por 2, e utilizando a notação científica
  - = 111011_2 * 2^-2
  - = 111011_2 * 2^-2 * 2^5 * 2^-5 (SHR)
  - = 1.11011_2 * 2^-2 * 2^5
  - = 1.11011_2 * 2^3 expoente
    mantissa

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Passando 1.11011_2 para decimal
  - 2^0 . 2^{-1} 2^{-2} 2^{-3} 2^{-4} 2^{-5}
  - 1 . 1 1 0 1 1
  - = 1*2^0 + 1*2^-1 + 1*2^-2 + 0*2^-3 + 1*2^-4 + 1*2^-5
  - = 1.84375

    mantissa

Tipos básicos de dados

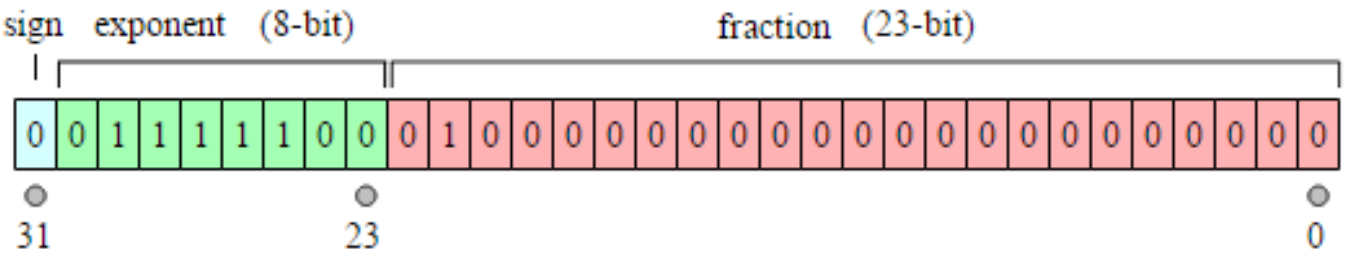
- Ponto flutuante (float/double)
  - = 1.11011_2 * 2^3
  - = 1.84375 * 2^3

  - Verificando a conta acima:
  - = 1.84375 * 2^3
  - = 14.75

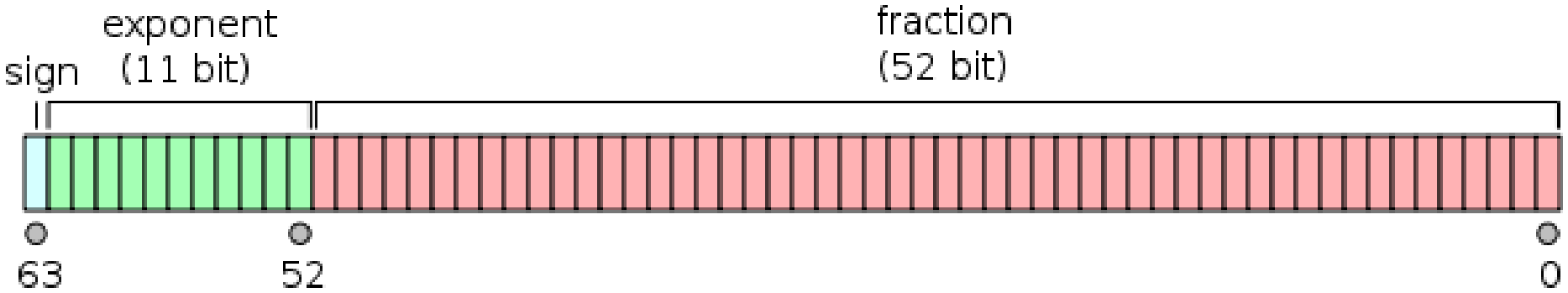
Tipos básicos de dados

- Ponto flutuante (float/double)
  - Representação padronizada – IEEE 754

float



double



  - Expoente representado com bias 127 (variação -127 ~ +128 para float)

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Exemplo anterior
  - = 1.11011_2 * 2^3
  - Bits da fração: 11011
  - Expoente com bias = 127 + 3 = 130 = 10000010_2
  - Sinal = 0 (positivo)

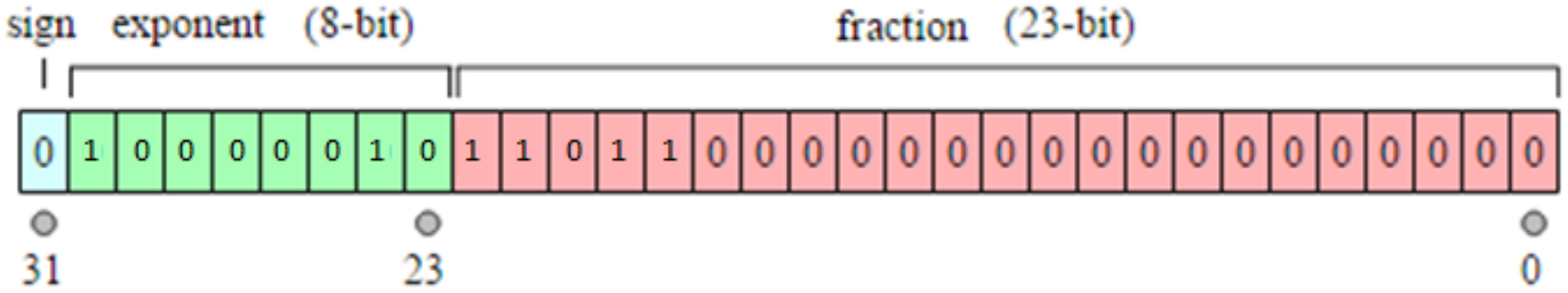
Tipos básicos de dados

- Ponto flutuante (float/double)
  - Verificando o exemplo em código

```
#include <stdio.h>
#include <string.h>
#include <assert.h>

int main()
{
   float f  valor = 14.75;
   int i_valor = 0;

   assert(sizeof(int) == sizeof(float));
   memcpy(&i_valor, &f_valor, sizeof(int));

   printf("%X \n", i_valor);

   return 0;
}
```

Tipos básicos de dados

- Ponto flutuante (float/double)
  - Esperado: 0x416C0000



Tipos básicos de dados

- Vazio (void)
  - Usado pra definir ausência de tipo em parâmetros, retornos ou ponteiros
  - void func(...);
  - int func(void);
  - void *p;

Tipos básicos de dados

- Tipos definidos pelo usuário
- Tipos derivados

- Aula específica no futuro!

Modificadores de tipos

- Tamanho
  - short int – maior ou igual a 16 bits, maior ou igual a char
  - long int – maior ou igual a 32 bits, maior ou igual a int
  - long double – maior ou igual a double

Modificadores de tipos

- Sinal
  - signed – Permite valores negativos (padrão)
  - unsigned – Apenas valores positivos

Modificadores de tipos

- Sinal (signed/unsigned)
  - Como representar valores negativos?
  - Magnitude assinalada
  - Complemento de um
  - Complemento de dois

Modificadores de tipos

- Sinal (signed/unsigned)
  - Complemento de dois – amplamente utilizado em várias arquiteturas
  - Para representar -1, considera-se o valor como positivo, considerando uma palavra de 4 bits:
  - 0001, aplica-se a negação lógica:
  - 1110, em seguida soma-se um:
  - 1111
  - ^ O primeiro bit representa o sinal

Modificadores de tipos

- Constante (const)
  - Variáveis cujo conteúdo não pode ser alterado após a declaração

```
const int A = 10;
const unsigned B = 20;
```

Modificadores de tipos

- Armazenamento
  - Define o escopo (visibibilidade) e duração das variáveis
  - auto – variáveis na pilha. (padrão)
  - register – variáveis em registradores.
  - static – variável é mantida durante toda a existência do programa, ao contrário de ser criada e destruída cada vez que o escopo é acessado. nomes simbólicos globais são limitados ao escopo do arquivo.
  - extern – nomes simbólicos são visíveis a todos os arquivos de código. (padrão)
  - Ponteiros, funções e interrupções – aula futura.

Literais

- 0x... – hexadecimal
- 0... – octal
- 5u – unsigned
- 5l - long
- 5ul – unsigned long
- 314159E-5L – notação exponencial

limits.h

- Macros que definem limites dos tipos

```
#include <stdio.h>
#include <limits.h>

int main()
{

  printf("Numero de bits em um byte %d\n", CHAR_BIT);

  printf("Valor mínimo de SIGNED CHAR = %d\n". SCHAR  MIN):
  printf("Valor máximo de SIGNED CHAR = %d\n". SCHAR  MAX):
  printf("Valor máximo de UNSIGNED CHAR = %d\n", UCHAR_MAX);

  printf("Valor mínimo de SHORT INT = %d\n". SHRT  MIN):
  printf("Valor mínimo de SHORT INT = %d\n", SHRT_MAX);

  printf("Valor mínimo de INT = %d\n". INT  MIN):
  printf("Valor mínimo de INT = %d\n", INT_MAX);

  printf("Valor mínimo de CHAR = %d\n". CHAR  MIN):
  printf("Valor máximo de CHAR = %d\n", CHAR_MAX);

  printf("Valor mínimo de LONG = %ld\n". LONG  MIN):
  printf("Valor máximo de LONG = %ld\n", LONG_MAX);

  return(0);
}
```