PARADIGMAS E LINGUAGENS DE PROGRAMAÇÃO

ENGENHARIA DA COMPUTAÇÃO – UFC/SOBRAL

Prof. Danilo Alves

danilo.alves@alu.ufc.br

INTRODUÇÃO

UNIVERSIDADE FEDERAL DO CEARÁ

- Um tipo de dados define uma coleção de valores de dados e um conjunto de operações pré-definidas sobre eles
- Um descritor é a coleção de atributos de uma variável
  - Nome, endereço, valor, tipo, tempo de vida, escopo
- Uma questão de projeto para todos os tipos de dados: Que operações são fornecidas para variáveis do tipo e como elas são especificadas?

TIPOS DE DADOS PRIMITIVOS

UNIVERSIDADE FEDERAL DO CEARÁ

- Praticamente todas as linguagens de programação fornecem um conjunto de tipos de dados primitivos
- Tipos de dados primitivos: Aqueles não definidos em termos de outros tipos
  - Ex.: inteiro, real, caractere

TIPOS DE DADOS PRIMITIVOS

UNIVERSIDADE FEDERAL DO CEARÁ

- Alguns dos tipos primitivos são meramente reflexos de hardware
  - Ex.: int, em C

```
int a = 12345678945678945612345612345679887894561 23;
warning: integer constant is too large for its type
     int a = 12345678945678945612345612345679887894561 23;
                ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
warning: overflow in implicit constant conversion

Valor atribuído: -522375941 (se inteiro de 4 bytes)
```

- Outros requerem apenas um pouco de suporte externo ao hardware para sua implementação
- – Ex.: inteiro longo (long) em Python 2

TIPOS DE DADOS PRIMITIVOS: INTEIRO

UNIVERSIDADE FEDERAL DO CEARÁ

- Quase sempre um reflexo exato do hardware, logo o mapeamento é simples
- Muitos computadores suportam diversos tipos de tamanhos inteiros
- Java inclui quatro tamanhos inteiros com sinal:
  - byte, short, int, long

TIPOS DE DADOS PRIMITIVOS: INTEIRO

UNIVERSIDADE FEDERAL DO CEARÁ

byte	8	-128	127
short	16	-32768	32767
int	32	-2.147.483.648	2.147.483.647
long	64	-9223372036854775808	9223372036854775807

TIPOS DE DADOS

UNIVERSIDADE FEDERAL DO CEARÁ

Tipo	Tamanho em bits	Valores	Padrão
`boolean`		`true` ou `false`	
[*Observação*: a representação de um `boolean` é específica à Java Virtual Machine em cada plataforma.]			
`char`	16	`'\u0000'` a `'\uFFFF'` (0 a 65535)	(conjunto de caracteres Unicode ISO)
`byte`	8	–128 a +127 (-2^7 a $2^7 - 1$)	
`short`	16	–32.768 a +32.767 (-2^{15} a $2^{15} - 1$)	
`int`	32	–2.147.483.648 a +2.147.483.647 (-2^{31} a $2^{31} - 1$)	
`long`	64	–9.223.372.036.854.775.808 a +9.223.372.036.854.775.807 (-2^{63} a $2^{63} - 1$)	
`float`	32	*Intervalo negativo:* –3,4028234663852886E+38 a –1,40129846432481707e–45 *Intervalo positivo:* 1,40129846432481707e–45 a 3,4028234663852886E+38	(IEEE 754, ponto flutuante)
`double`	64	*Intervalo negativo:* –1,7976931348623157E+308 a –4,94065645841246544e–324 *Intervalo positivo:* 4,94065645841246544e–324 a 1,7976931348623157E+308	(IEEE 754, ponto flutuante)

TIPOS DE DADOS PRIMITIVOS: PONTO FLUTUANTE

UNIVERSIDADE FEDERAL DO CEARÁ

- **Modelam** números reais, mas as representações são apenas **aproximações**

  – Nenhum dos números fundamentais π ou e (base para logaritmos naturais) podem ser corretamente representados em notação de ponto flutuante ou em qualquer espaço finito

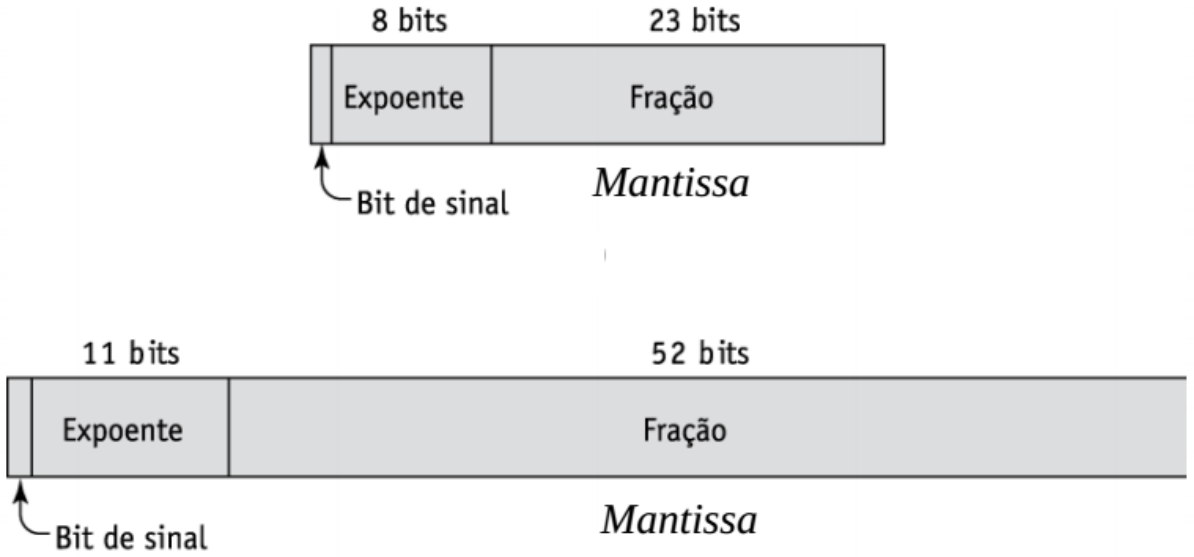
  – O problema é agravado quando representados em um computador é em binário

  – Ex.: 0.1 (em decimal) é equivalente a 0.0001100110011... (em binário)

- Linguagens para uso científico suportam pelo menos dois tipos de ponto flutuante (por exemplo, **float** e **double**)

TIPOS DE DADOS PRIMITIVOS: PONTO FLUTUANTE

- Linguagens atuais usam o padrão **IEEE Padrão de Ponto Flutuante 754**, da IEEE



TIPOS DE DADOS PRIMITIVOS: COMPLEXO

UNIVERSIDADE FEDERAL DO CEARÁ

- Algumas linguagens suportam um tipo de dados complexo – por exemplo, Fortran e Python
- Valores complexos são representados como pares ordenados de valores de ponto flutuante
- Literal complexo (em Python):
  - (7 + 3j), onde 7 é a parte real e 3 é a parte imaginária
- Linguagens que suportam um tipo complexo incluem operações para aritmética em valores complexos.

TIPOS DE DADOS PRIMITIVOS: COMPLEXO

UNIVERSIDADE FEDERAL DO CEARÁ

- Soma e subtração de números complexos são feitos de forma isolada para cada parte

```
a = (7 + 3j)  # j é equivalente a sqrt(-1)
b = (4 + 7j)

c = a + b  # c = (11 + 10j)

a * b      # 7x4 + (7*7)j +(3*4)j + (3x7)j²  = (7 + 61j)
           #                          j²  = (-1 + 0j)
```

- Conjugado de um número

```
z = (7 + 3j)
z.conjugate()   # retorna (7 - 3j)
```

TIPOS DE DADOS PRIMITIVOS: COMPLEXO

UNIVERSIDADE FEDERAL DO CEARÁ

```c
#include <stdio.h>
#include <complex.h>

int main(int argc, char const *argv[]) {
    _Complex z = 2 + 3 * I;

    printf("%.2lf+%.2lfi \n", creal(z), cimag(z));
    return 0;
}
```

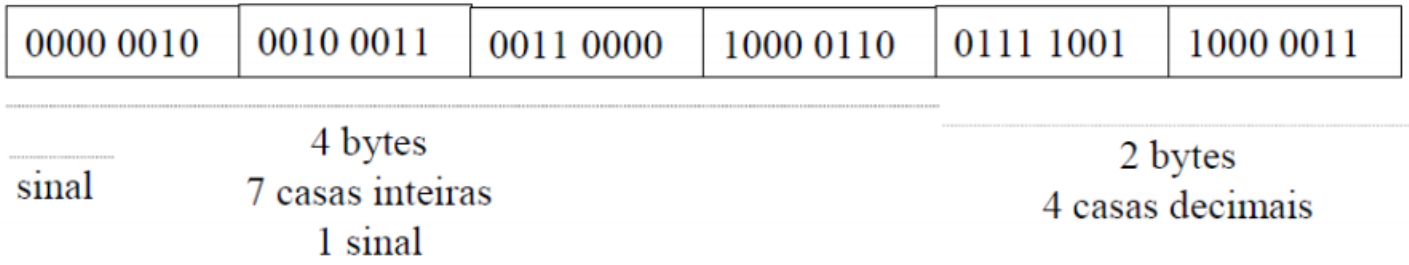
TIPOS DE DADOS PRIMITIVOS: COMPLEXO

UNIVERSIDADE FEDERAL DO CEARÁ

```c
#include <stdio.h>
#include <complex.h>

int main(int argc, char const *argv[]) {
    _Complex z = 2 + 3 * I;

    printf("%.2lf+%.2lfi \n", creal(z), cimag(z));

    printf(“Conjugado\n”);
    z = conj(z);
    printf("%.2lf+%.2lfi \n", creal(z), cimag(z));

    return 0;
}
```

TIPOS DE DADOS PRIMITIVOS: DECIMAL

- Para aplicações de sistemas de negócios
  - Essencial para COBOL
  - C# tem um tipo de dados decimal
  - Python implementa decimal por meio da classe **Decimal** do módulo **decimal**
- Decimais codificados em binário (BCD)
- Vantagem: precisão
- Desvantagens: faixa de valores restrita, gasto de memória



TIPOS DE DADOS PRIMITIVOS: BOOLEANOS

UNIVERSIDADE FEDERAL DO CEARÁ

- Mais simples de todos, devido seu pequeno domínio
- Faixa de valores: dois elementos, um para "verdadeiro" e um para "falso"
- Poderiam ser representados por bits, mas são armazenados em bytes
  - Vantagem: legibilidade
- Presente em linguagens como C++, Java, Python e C#
  - Ausente na primeira padronização de C – presente hoje através de **_Bool**
- Através da biblioteca *stdbool.h*, obtém-se o alias **bool** e os literais **true** e **false**
- Operações lógicas
  - and, or, not

TIPOS DE DADOS PRIMITIVOS: BOOLEANOS

UNIVERSIDADE FEDERAL DO CEARÁ

```c
#include <stdio.h>
#include <stdbool.h>

int main(void) {
    _Bool b = true;

    ...

    return 0;
}
```

```c
public class Main {
    public static void Main(string[] args) {
        bool x = false;

        System.Console.WriteLine(x);
    }
}
```

```c
public class Main {
    public static void main(String[] args) {
        boolean x = false;

        System.out.println(x);
    }
}
```

TIPOS DE DADOS PRIMITIVOS: BOOLEANOS

UNIVERSIDADE FEDERAL DO CEARÁ

- Python

```
x = False;
print(x);
```

- JavaScript

```
var x = false;
console.log(x);
```

- PHP

```
$x = false; // ou FALSE
echo($x);
```

TIPOS DE DADOS PRIMITIVOS: CARACTERE

UNIVERSIDADE FEDERAL DO CEARÁ

- Armazenados como codificações numéricas
- Codificação mais usada: ASCII
- Uma alternativa, conjunto de caracteres de 16 bits: Unicode (UCS-2)
  - Inclui caracteres da maioria das linguagens naturais
  - Originalmente usado em Java
  - C# JavaScript e Python também suportam Unicode
- Unicode 32 bits (UCS-4)
  - Suportada por Fortran, começando com 2003

TABELA ASC II

UNIVERSIDADE FEDERAL DO CEARÁ

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

OPERAÇÕES

UNIVERSIDADE FEDERAL DO CEARÁ

- **isalpha()** alfabético
- **isdigit()** dígito decimal
- **isalnum()** alfanumérico
- **isspace()** verifica se corresponde a um espaço
- **islower()** caixa baixa
- **isupper()** caixa alta
- **isxdigit()** é um dígito hexadecimal
- **iscntrl()** é caractere de controle \n, \t

tolower() converte para caixa baixa

toupper() converte para caixa alta

isgraph() é um gráfico

ispunct() é pontuação

isprint() é imprimível

CADEIAS DE CARACTERES

UNIVERSIDADE FEDERAL DO CEARÁ

- Valores são sequências de caracteres
- Questões de projeto:
  - As cadeias devem ser apenas um **tipo especial de vetor de caracteres** ou um **tipo primitivo**?
  - As cadeias devem ter tamanho estático ou dinâmico?
    - Em C, cadeias de caracteres têm tamanhos estáticos ou dinâmicos?
- Operações comuns:
  - Atribuição
  - Comparação (=, > etc.)
  - Concatenação
  - Referência a subcadeias

CADEIAS CADEIAS E SUAS OPERAÇÕES DE CARACTERES

UNIVERSIDADE FEDERAL DO CEARÁ

- Problema com *strcpy(dest, src)*, em C?

– Se o tamanho de *dest* é 20 e o de *src* é 50, *strcpy* escreverá sobre os 30 bytes subsequentes a *dest*

CADEIAS DE CARACTERES EM ALGUMAS LINGUAGENS

UNIVERSIDADE FEDERAL DO CEARÁ

- C e C++
  - Não são definidas como primitivas
  - Usam matrizes de caracteres char e uma biblioteca de funções que fornecem operações
- SNOBOL4
  - Primitivas
  - Várias operações
- Fortran e Python
  - Tipo primitivo com atribuição e diversas operações
- Java
  - Primitiva via classe String (C++ também implementa uma classe string)
- Perl, JavaScript, Ruby e PHP
  - Incluem operações padrões pré-definidas, usando expressões regulares

OPÇÕES DE TAMANHO DE CADEIAS

UNIVERSIDADE FEDERAL DO CEARÁ

- **Estático**: COBOL, classe pré-definida String

- **Dinâmico limitado**: C e C++

  – Nessas linguagens, um ponteiro pode representar o tamanho a ser alocado

- **Dinâmico**: SNOBOL4, Perl, JavaScript
- Ada 95 suporta as três opções

AVALIAÇÃO

UNIVERSIDADE FEDERAL DO CEARÁ

- Importantes para a facilidade de escrita
- Como tipos primitivos de tamanho estático não são custosos, por que não tê-los em uma linguagem? Tamanho dinâmico é interessante, mas vale o custo?

IMPLEMENTAÇÃO DE CADEIAS DE CARACTERES

UNIVERSIDADE FEDERAL DO CEARÁ

- **Tamanho estático**: descrito em tempo de compilação
- **Tamanho dinâmico limitado**: pode necessitar de um descritor em tempo de execução (mas não em C e C++, pois usam um ponteiro para demarcar o fim)
- **Tamanho dinâmico**: necessita de descritor em tempo de execução; alocação/liberação é o maior problema de implementação

TIPOS ORDINAIS DEFINIDOS PELO USUÁRIO

UNIVERSIDADE FEDERAL DO CEARÁ

- Um **tipo ordinal** é um no qual a faixa de valores possíveis pode ser facilmente associada com o **conjunto de inteiros positivos**
- Exemplos de tipos primitivos ordinais em Java
  - Integer
  - char
  - Boolean
- Existem dois tipos ordinais definidos pelo usuário
  - Tipo **enumeração**
  - Tipo **subfaixa**

TIPOS ENUMERAÇÃO

UNIVERSIDADE FEDERAL DO CEARÁ

- Todos os valores possíveis, os quais são constantes nomeadas, são fornecidos na definição

- Exemplo em C, C++, C#, Java
  - **enum** dias {DOM, SEG, TER, QUA, QUI, SEX, SAB};

- C/C++, cada elemento corresponde a um inteiro, podendo ter valores alterados
  - **enum** dias var = SEG;

- C#/Java, os elementos não são convertidos para um inteiro
  - Dias d = Dia.SEG;

TIPOS ENUMERAÇÃO

UNIVERSIDADE FEDERAL DO CEARÁ

- Exemplo em C, C++, C#, Java

  enum dias {DOM = 10, SEG, TER, QUA, QUI, SEX, SAB = 30};

- **enum** dias var = 10
  - válido para C, mas não para C++
  - C++ requer que se faça um cast para (dias)

PYTHON IMPLEMENTA ENUMERAÇÃO COM A CLASSE ENUM, DO MÓDULO ENUM

UNIVERSIDADE FEDERAL DO CEARÁ

- Python implementa enumeração com a classe Enum, do módulo enum

```
from enum import Enum

class Mes(Enum):
    JAN = 1
    FEV = 2
    MAR = 3
    ABR = 4
    MAI = 5
    JUN = 6
    JUL = 7
    AGO = 8
    SET = 9
    OUT = 10
    NOV = 11
    DEZ = 12
```

```
for mes in Mes:
    print(mes.name, mes.value)


mes = Mes(6)
print(mes)     # Mes.JUN

mes = Mes.JUN
print(mes)     # Mes.JUN
```

TIPOS ENUMERAÇÃO

UNIVERSIDADE FEDERAL DO CEARÁ

- Questões de projeto
  - Uma constante de enumeração pode aparecer em mais de uma definição de tipo?
  - Os valores de enumeração são convertidos para inteiros?
  - Existem outros tipos que são convertidos para um tipo enumeração?

AVALIAÇÃO DE TIPOS ENUMERAÇÃO

UNIVERSIDADE FEDERAL DO CEARÁ

- **Melhora a legibilidade**, por exemplo, não precisa codificar uma cor como um número

- **Melhora a confiabilidade**, por exemplo, compilador pode verificar:
  - Operações (não permitir que as cores sejam somadas)
  - Não se pode definir valores fora da faixa da enumeração
  - Ada, C# e Java >= 5.0 não fazem correção para inteiro

TIPOS SUBFAIXA

UNIVERSIDADE FEDERAL DO CEARÁ

- Uma subsequência contígua de um tipo ordinal
  - Exemplo: 12..18 é uma subfaixa do tipo inteiro
- Projeto de Ada
  - **type** Days **is** (mon, tue, wed, thu, fri, sat, sun);
  - **subtype** Weekdays is Days range mon..fri;
  - **subtype** Index is Integer range 1..100;

    Day1: Days;

    Day2: Weekday;

    Day2 := Day1;
  - A atribuição é legal?

AVALIAÇÃO DO TIPO SUBFAIXA

UNIVERSIDADE FEDERAL DO CEARÁ

- Melhora a legibilidade

  – Podem armazenar apenas certas faixas de valores

- Melhora a confiabilidade

  – A atribuição de um valor a uma variável de subfaixa que está fora da faixa especificada é detectada como um erro

IMPLEMENTAÇÃO DE TIPOS ORDINAIS DEFINIDOS PELO USUÁRIO

UNIVERSIDADE FEDERAL DO CEARÁ

- Tipos enumeração **geralmente** são implementados como inteiros

- Tipos subfaixas são implementados como seus tipos ancestrais

  – Verificações de faixas devem ser incluídas 3 em cada atribuição de uma variável ou de uma expressão a uma variável subfaixa