ROTEIRO

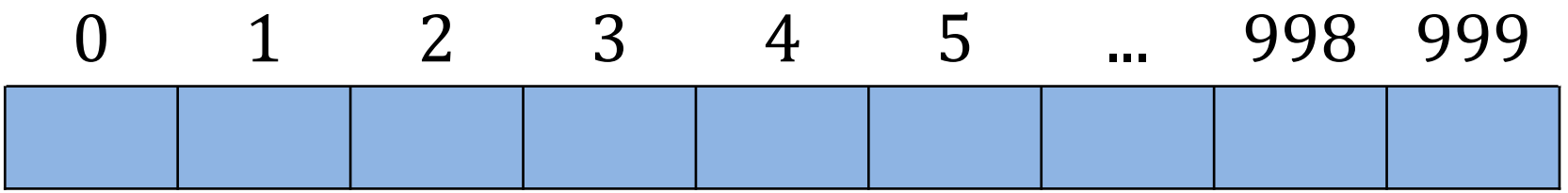
1. Strings
2. Biblioteca string.h
3. Ponteiros

VETORES (ARRAY/ARRANJOS)

- Arrays são agrupamentos de dados adjacentes na memória do mesmo tipo.
- Vetor é um ***array de uma dimensão.***
- Declaração:
  tipo_dado *nome_array*[tamanho];

O comando acima define um array de nome *nome_array*, capaz de armazenar tamanho elementos adjacentes na memória do tipo tipo_dado

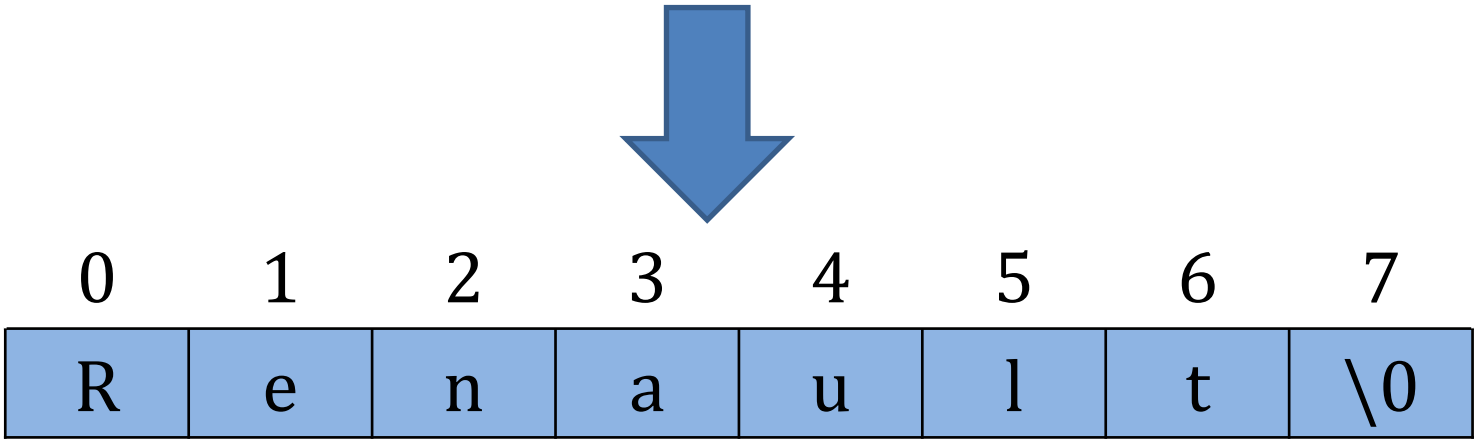
- Ex: float velocidades[1000];
- int notas[100];



REPRESENTANDO PALAVRAS

- Vetor de caracteres
- `char palavra[tamanho];`

- `char marca[8] = "Renault";`



STRING (CORDA)

- Chama-se um vetor de caracteres de *String* (corda)
- Também conhecido como literais e são representados por caracteres entre aspas duplas
  - `"Engenharias"`
  - `"Velocidade Máxima"`

Fonte: www.freeimageslive.co.uk

CARACTERES ASCII

Fonte: https://www.techtudo.com.br/noticias/noticia/2015/02/o-que-e-o-codigo-ascii-e-para-que-serve-descubra.html

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

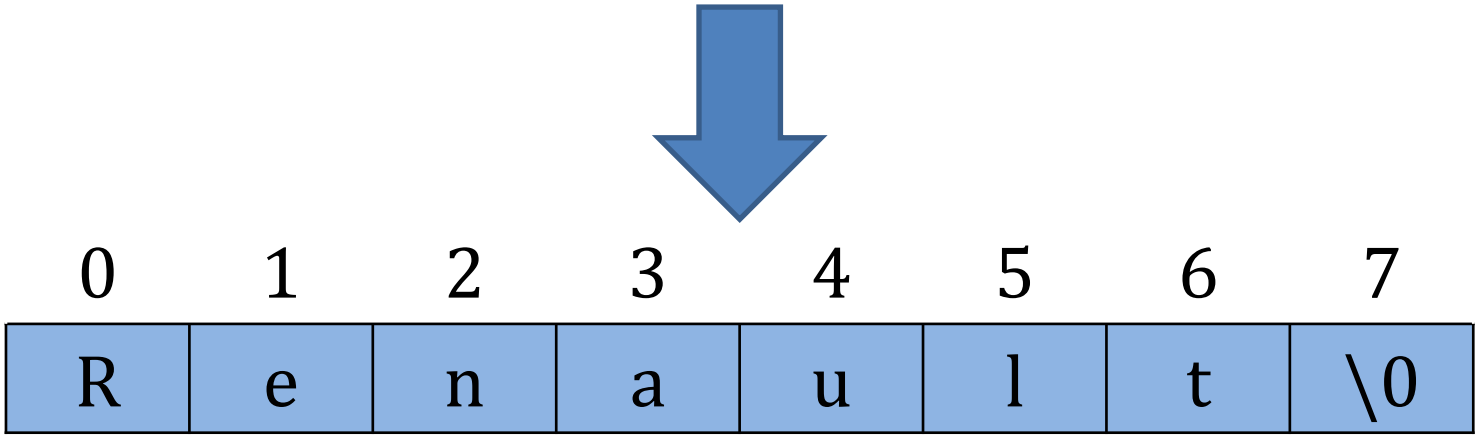
STRING (CORDA)

- Um vetor deve ter tamanho fixo
- Como armazenar textos com tamanhos diferentes?
  - Marcas de veículos
  - `Nome de pessoas`
  - `Locais atendidos`

STRING

- Como saber onde uma string termina?
- Toda string deve ser terminada pelo caractere nulo (`\0`)

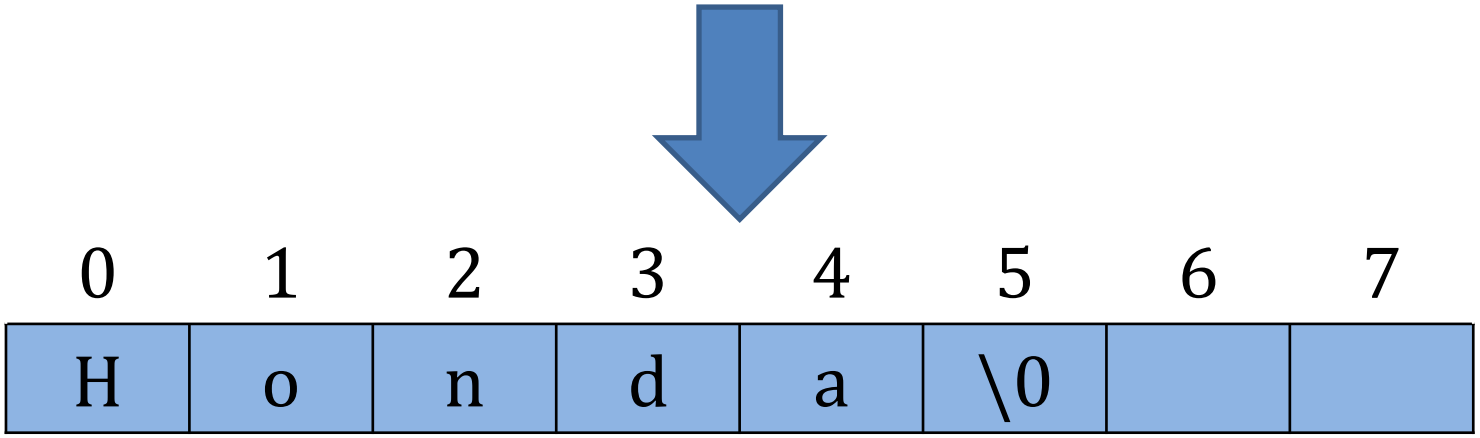
```
char marca[8] = "Renault";
```



STRING

- Como saber onde uma string termina?
- Toda string deve ser terminada pelo caractere nulo (`\0`)

```
char marca[8] = "Honda";
```



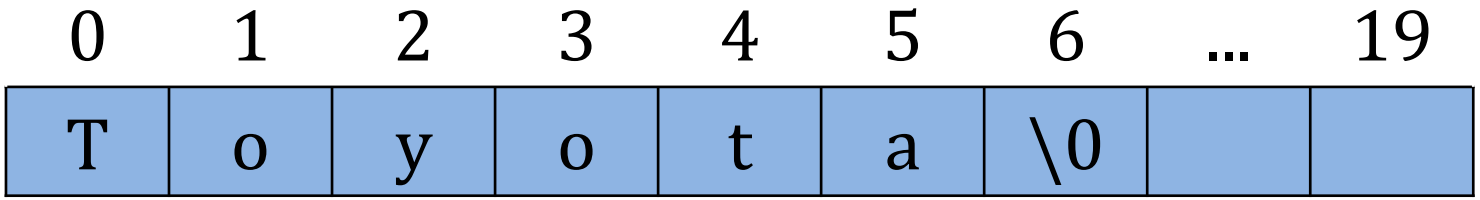
STRING

```
char marca[8] = "Honda";
```

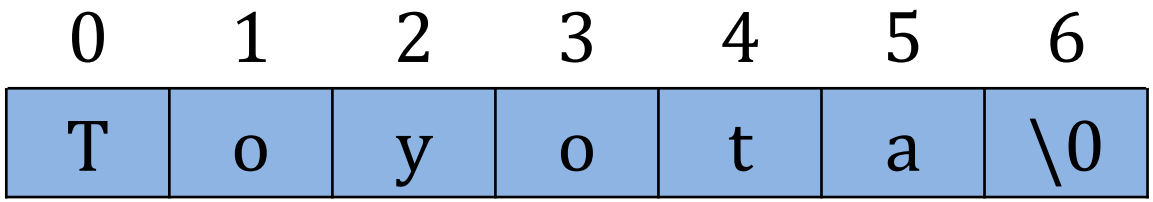
- Note que essa declaração é equivalente a:
- `char marca[0] = 'H'`
- `char marca[1] = 'o'`
- `char marca[2] = 'n'`
- `char marca[3] = 'd'`
- `char marca[4] = 'a'`
- `char marca[5] = '\0'`

INICIALIZAÇÃO DE STRINGS

- `char marca[20] = "Toyota";`



- `char marca[] = "Toyota";`



INICIALIZAÇÃO DE STRINGS

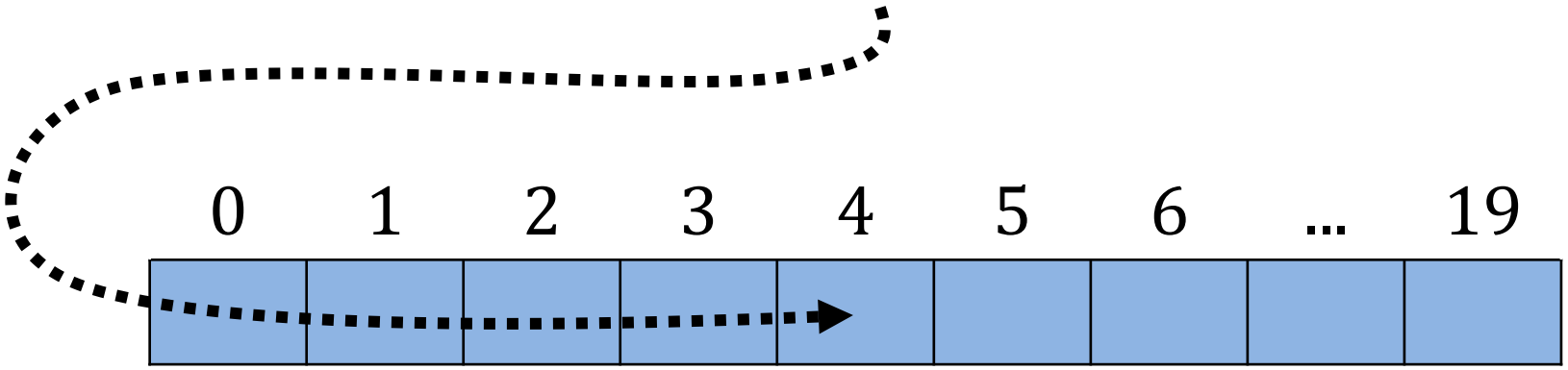
- Uma string pode ser lida através de várias funções

- **scanf: scan formatted**
- **gets: get string**
- **fgets: get fixed amount of characters**

LEITURA (SCANF)

- Podemos ler uma string caractere a caractere, mas é mais simples lê-la de uma vez.

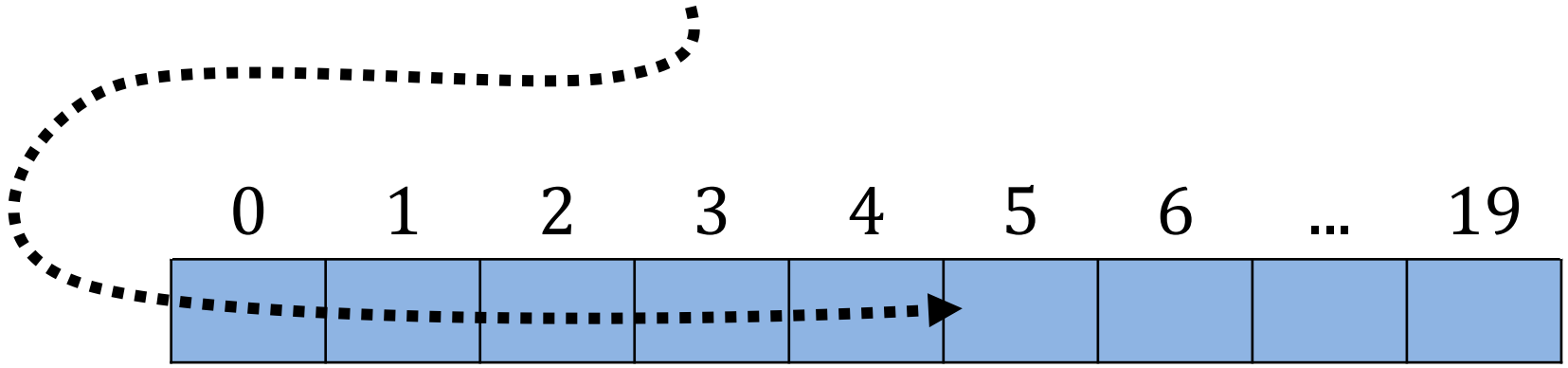
- `char marca[20];`
- `scanf("%s", &marca[0]);`



LEITURA (SCANF)

- Podemos ler uma string caractere a caractere, mas é mais simples lê-la de uma vez.

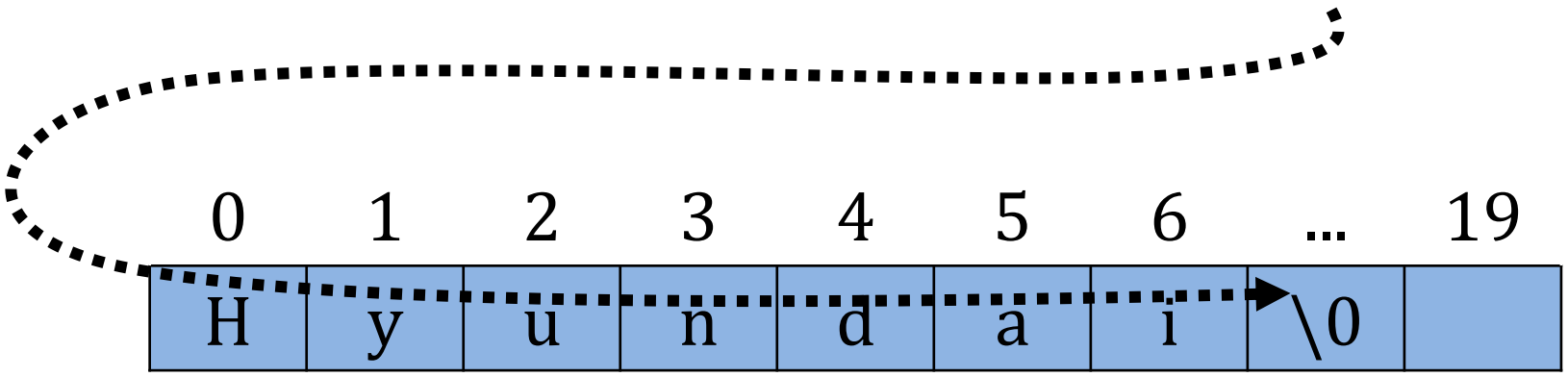
- `char marca[20];`
- `scanf("%s", marca);`



ESCRITA (PRINTF)

- Podemos imprimir uma string caractere a caractere, mas é mais simples imprimi-la de uma vez.

- `char marca[20] = "Hyundai";`
- `printf("Marca do Veículo: %s", marca);`



LEITURA E ESCRITA (SCANF / PRINTF)

```c
#include <stdio.h>

int main() {
    char marca[10];

    scanf("%s", marca);

    printf("Marca lida %s", marca);
}
```

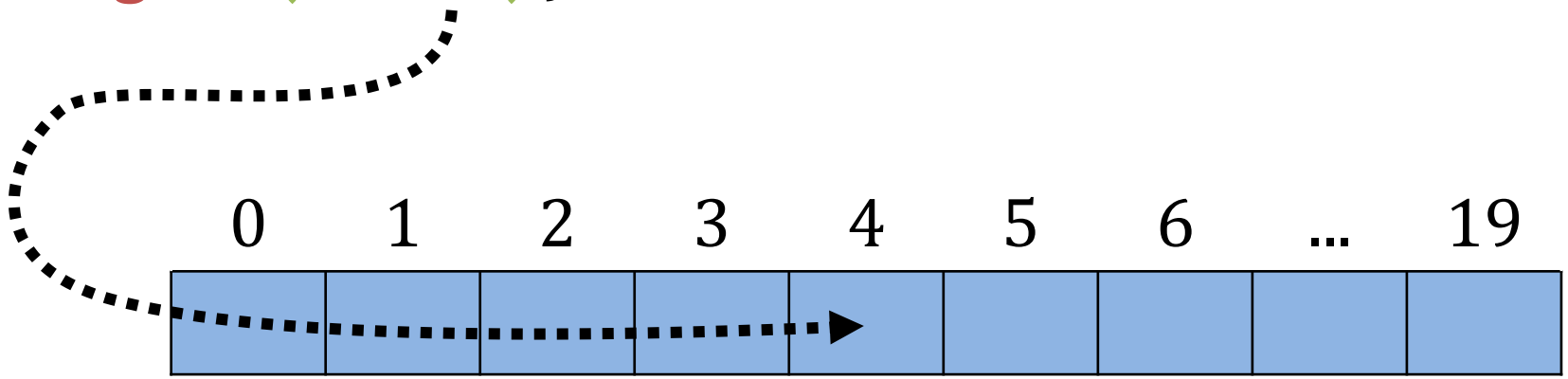
LEITURA (SCANF)

- Lê somente até o primeiro espaço
- Contornar isso, modificar o formato no scanf, utilizar outras funções (gets e fgets).
- Outra função para imprimir: puts (put string)

LEITURA (GETS)

- A função gets (get string) faz a leitura até encontrar o caractere de fim de linha (enter)

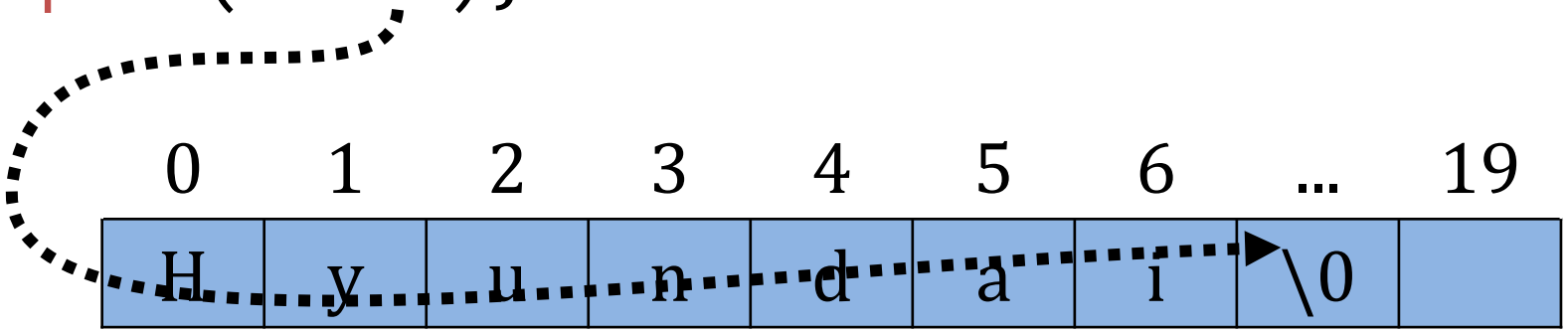
- `char marca[20];`
- `gets(marca);`



ESCRITA (PUTS)

- Semelhante a gets (get string), a função puts (put string) faz a escrita da string até encontrar o caractere de fim de linha (enter)

- `char marca[20] = "Hyundai";`
- `puts(marca);`

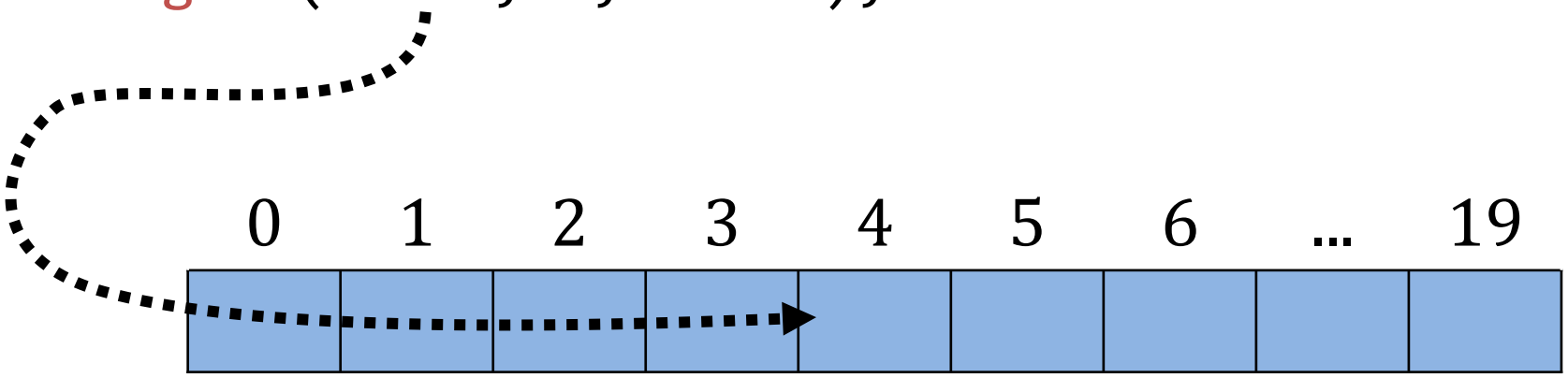


LEITURA E ESCRITA (GETS/PUTS)

```
 1  #include <stdio.h>
 2
 3  int main() {
 4      char marca[10];
 5
 6      gets(marca);
 7
 8      puts("Marca Lida: ");
 9      puts(marca);
10  }
```

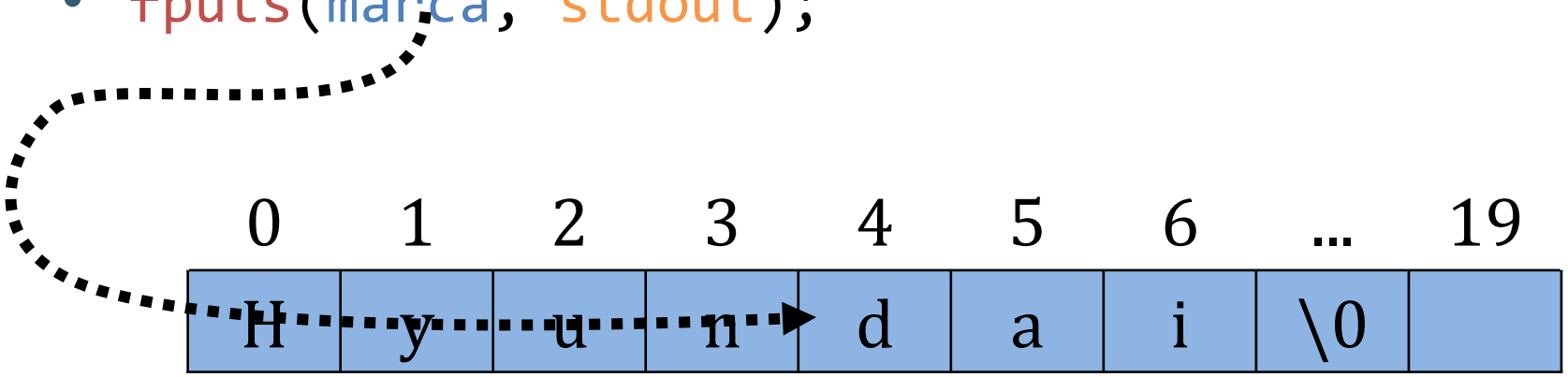
LEITURA (FGETS)

- Semelhante a gets mas limita a leitura a n caracteres
- fgets(varstring, n, stdin)
  - varstring: vetor de char criado
  - n: tamanho máximo a ser lido
  - stdin: indica que a leitura será feita pelo teclado
  - fgets também pode ler strings de arquivos
- `char marca[20];`
- `fgets(marca, n, stdin);`



ESCRITA (FPUTS)

- Semelhante a puts mas pode escrever a string em outros locais
- fputs(varstring, stdout)
  - varstring: vetor de char criado
  - stdout: indica que a escrita será feita na tela
  - fputs também pode escrever strings em arquivos
- `char marca[20] = "Hyundai";`
- `fputs(marca, stdout);`



LEITURA E ESCRITA (FGETS/FPUTS)

```
1   #include <stdio.h>
2
3   int main() {
4       char marca[10];
5
6       fgets(marca, 10, stdin);
7
8       fputs("Marca Lida: ", stdout);
9       fputs(marca, stdout);
10  }
```

MANIPULAÇÃO DE STRING

```
char marca[8] = "Honda";
```

- Note que essa declaração é equivalente a:
- `char marca[0] = 'H'`
- `char marca[1] = 'o'`
- `char marca[2] = 'n'`
- `char marca[3] = 'd'`
- `char marca[4] = 'a'`
- `char marca[5] = '\0'`

MANIPULAÇÃO DE STRING

- String é um tipo de dado complexo (vetor de char)
- A string (vetor) como um todo não pode ser manipulado igual aos tipos primitivos

- Não se pode fazer isso:

```
1. char a[20] = "Acre";
2. char b[20] = "Bahia";
3. b = a;
```

MANIPULAÇÃO DE STRING

- String é tão importante que existe uma biblioteca inteira para manipulação desse tipo de dado
  - Tamanho da string
  - Concatenar strings
  - Procurar um caractere
  - Copiar string
  - Comparar

- Biblioteca: string.h
- #include ‹string.h›

BIBLIOTECA STRING.H: STRLEN

- #include ‹string.h›
- Função strlen(s)

- strlen retornar o comprimento da string s

- Exemplo:

1. char marca[20] = "Hyundai";
2. int x = strlen(marca);

BIBLIOTECA STRING.H: STRLEN

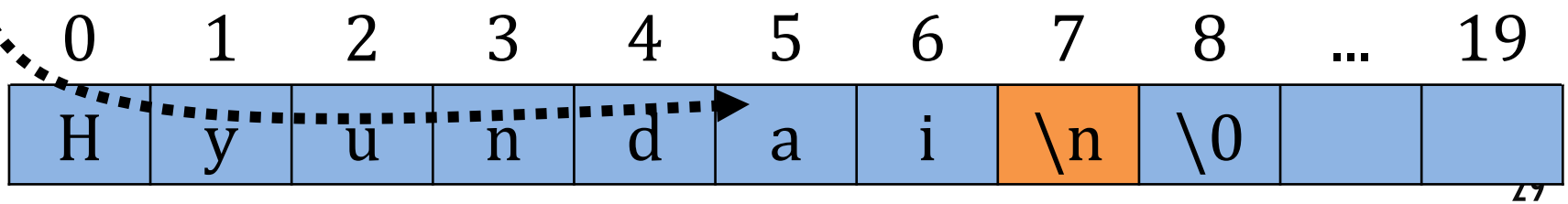
```
1   #include <stdio.h>
2   #include <string.h>
3
4   int main() {
5       char marca[20] = "Hyundai";
6       int x = strlen(marca);
7       printf("%d", x);
8   }
```

BIBLIOTECA STRING.H: STRLEN

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

int main() {
    char marca[20];
    fgets(marca, 20, stdin);
    int x = strlen(marca);
    printf("%d", x);
}
```

BIBLIOTECA STRING.H: STRLEN

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

int main() {
    char marca[20];
    fgets(marca, 20, stdin);
    int x = strlen(marca);
    printf("%d", x);
}
```



BIBLIOTECA STRING.H: STRCMP

- #include ‹string.h›
- função strcmp(s1, s2)
- strcmp compara as strings s1 e s2

- Retorna 0 se as strings são iguais
- Retorna <0 se o primeiro caractere diferente entre as duas tem valor menor em s1
- Retorna >0 se o primeiro caractere diferente entre as duas tem valor menor em s2

BIBLIOTECA STRING.H: STRCMP

```
Retorna 0 se são iguais
Retorna <0 se s1 < s2
Retorna >0 se s2 < s1
```

- Exemplo:

```
1. char m1[20] = "Hyundai";
2. char m2[10] = "Toyota";
3. char m3[10] = "Hyundai";

4. int a = strcmp(m1, m2);
5. int b = strcmp(m2, m1);
6. int c = strcmp(m1, m3);
7. printf("%d, %d, %d", a, b, c);
```

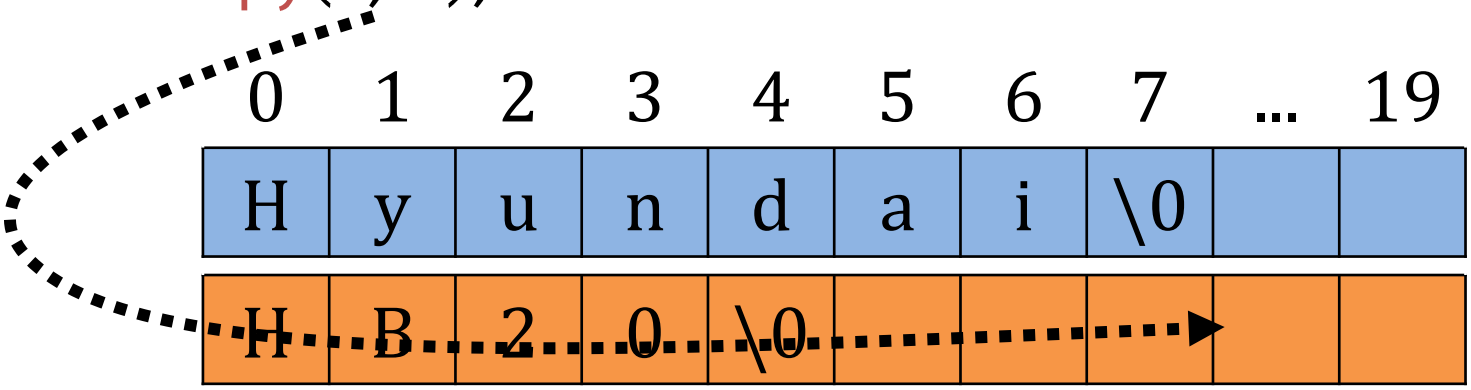
BIBLIOTECA STRING.H: STRCMP

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

int main() {
    char m1[20] = "Hyundai";
    char m2[10] = "Toyota";
    char m3[10] = "Hyundai";

    int a = strcmp(m1, m2);
    int b = strcmp(m2, m1);
    int c = strcmp(m1, m3);
    printf("%d, %d, %d", a, b, c);
}
```

BIBLIOTECA STRING.H: STRCPY

- #include ‹string.h›
- função strcpy(s1, s2)
- strcpy copia a string s2 para a string s1

- char a[20] = "Hyundai“;
- char b[20] = "HB20“;
- strcpy(a, b);



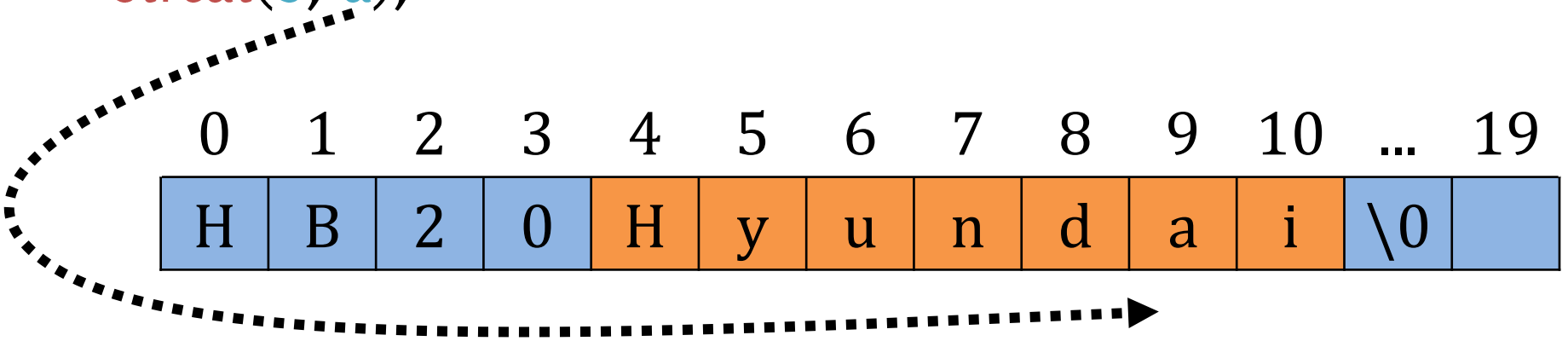
BIBLIOTECA STRING.H: STRCPY

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

int main() {
    char a[20] = "Hyundai";
    char b[20] = "HB20";
    strcpy(a, b);
    printf("%s", a);
}
```

BIBLIOTECA STRING.H: STRCAT

- #include ‹string.h›
- função strcat(s1, s2)
- strcat concatena a string s2 na string s1

- char a[20] = "Hyundai“;
- char b[20] = "HB20“;
- strcat(b, a);



OUTRAS FUNÇÕES

Outras funções:

- strncat: Semelhante a strcat mas especifica usa um inteiro para indicar substring a ser copiada
- strchr: encontrar primeira ocorrência de char
- strtook : divide string em substrings

http://www.cplusplus.com/reference/cstring/

EXERCÍCIOS

https://www.urionlinejudge.com.br/judge/pt/problems/view/2483
https://www.urionlinejudge.com.br/judge/pt/problems/view/2304
https://www.urionlinejudge.com.br/judge/pt/problems/view/2850

Desafios:
https://www.urionlinejudge.com.br/judge/pt/problems/view/2880
https://www.urionlinejudge.com.br/judge/pt/problems/view/2484

https://br.spoj.com/problems/JVESTI08/

JVESTI08 - Vestibular

A maioria das universidades brasileiras usa o *vestibular* para selecionar seus alunos. O vestibular consiste de uma ou mais provas sobre as matérias do Ensino Médio, visando avaliar os conhecimentos dos candidatos. Um formato popular de prova de vestibular é a *prova objetiva*. Neste formato de prova, cada candidato deve escolher uma das cinco alternativas apresentadas pela questão como sendo a correta. Durante a correção dos cartões, cada questão onde a alternativa escolhida pelo candidato é a mesma do gabarito, ele ganha um ponto. Alguns dos vestibulares mais concorridos do Brasil são disputados por dezenas de milhares de candidatos, e, por isso, geralmente usa-se uma folha de leitura óptica e um programa de computador para corrigir as provas de todos os candidatos e gerar a lista com suas pontuações. Você trabalha no comitê responsável pelo vestibular em uma faculdade respostas de um dos candidatos, determina o número de acertos daquele candidato.

Entrada

https://br.spoj.com/problems/AUTO08/

AUTO08 - Auto Estrada

Certas regiões resolveram o problema de tráfego intenso com a construção de auto estradas, que são estradas contendo em geral quatro ou mais pistas de rolagem em cada sentido, de forma que um número grande de carros possa passar sem que ocorram congestionamentos. O problema das auto estradas é que, junto com os carros temos um aumento considerável de ruído nas imediações da pista, o que incomoda os moradores das regiões próximas.

A GoTo engenharia, uma empresa do ramo de construção especializada em obras de estradas, encontrou uma solução engenhosa para o problema: instalar grandes painéis defletores de som de cada lado da auto estrada para tentar minimizar o ruído pe

Os painéis são construídos em blocos contínuos de 10 metros lineares. A auto estra extensão, sendo cada bloco descrito por um código, como definido abaixo:

• P - Pista, trecho em linha reta sem curvas ou saídas. Deve-se instalar um painel de cada

• C - Curva, trecho em curva de 90 graus na auto estrada. Deve-se instalar dois painéis de sem painel instalado.

• A - Acesso, trecho em linha reta no qual existe uma entrada ou uma saída a partir de Deve-se instalar um painel no lado onde não existe o acesso.

• D - Duplo acesso, trecho em linha reta no qual existem dois acessos (entradas ou saí lado da rodovia. Nenhum painel deve ser instalado nesse bloco da auto estrada.

https://br.spoj.com/problems/AUTO08/

AUTO08 - Auto Estrada

Certas regiões resolveram o problema de tráfego intenso com a construção de auto estradas, que são estradas contendo em geral quatro ou mais pistas de rolagem em cada sentido, de forma que um número grande de carros possa passar sem que ocorram congestionamentos. O problema das auto estradas é que, junto com os carros temos um aumento considerável de ruído nas imediações da pista, o que incomoda os moradores das regiões próximas.

A GoTo engenharia, uma empresa do ramo de construção especializada em obras de estradas, encontrou uma solução engenhosa para o problema: instalar grandes painéis defletores de som de cada lado da auto estrada para tentar minimizar o ruído pe

Os painéis são construídos em blocos contínuos de 10 metros lineares. A auto estra extensão, sendo cada bloco descrito por um código, como definido abaixo:

• P - Pista, trecho em linha reta sem curvas ou saídas. Deve-se instalar um painel de cada

• C - Curva, trecho em curva de 90 graus na auto estrada. Deve-se instalar dois painéis de sem painel instalado.

• A - Acesso, trecho em linha reta no qual existe uma entrada ou uma saída a partir de Deve-se instalar um painel no lado onde não existe o acesso.

• D - Duplo acesso, trecho em linha reta no qual existem dois acessos (entradas ou saí lado da rodovia. Nenhum painel deve ser instalado nesse bloco da auto estrada.