Universidade Federal do Rio Grande do Norte
Instituto Metrópole Digital

AULA 09

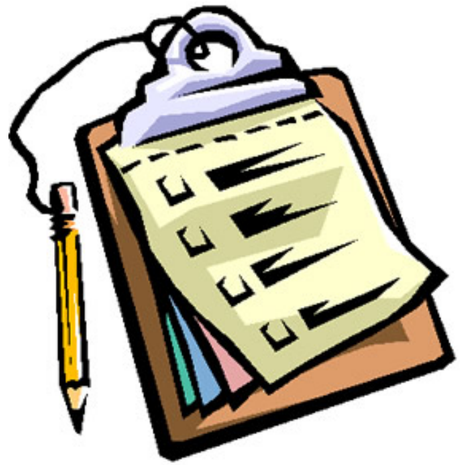
Estrutura de dados básico I (EDB1)

Prof. Msc. Janiheryson Felipe (Felipe)

Natal, RN
2023

OBJETIVO DA AULA

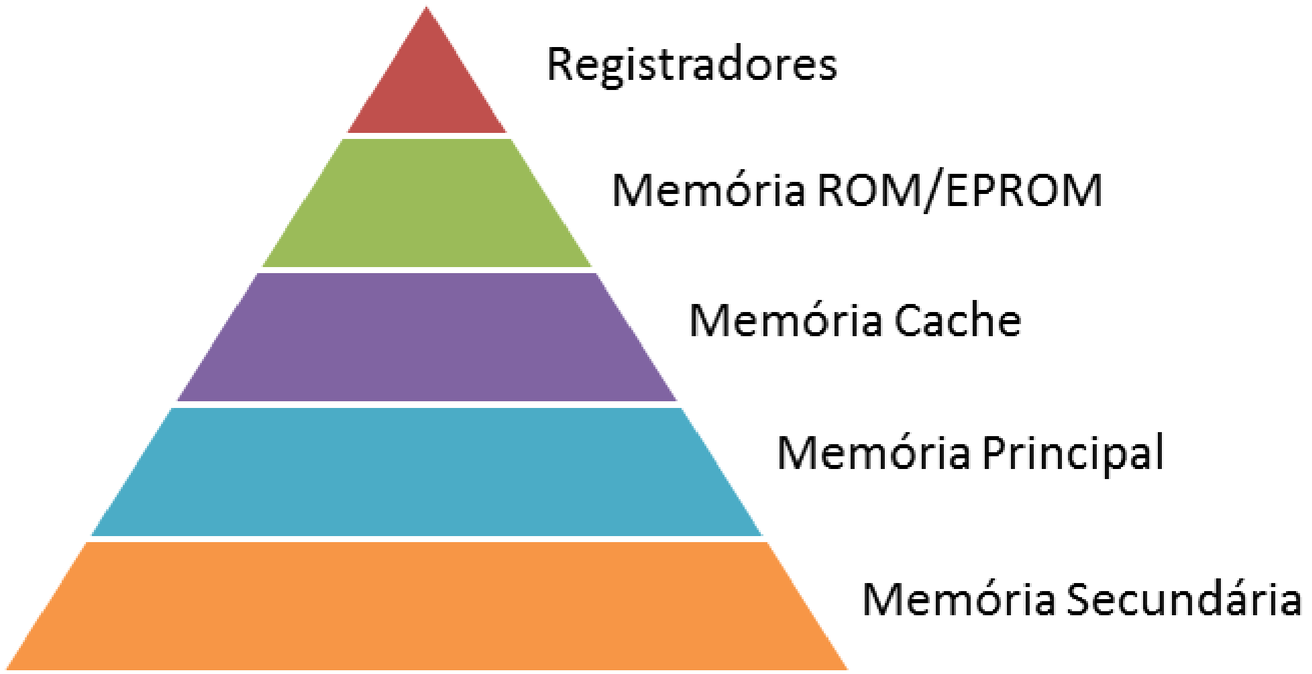
- Apresentar os tipos de dados e uso de memória pelos programas:
  - Tipos escalares
  - Tipos ponteiros
  - Tipos agregados (compostos)
  - Uso de memória pelos programas



ENTENDENDO A MEMÓRIA

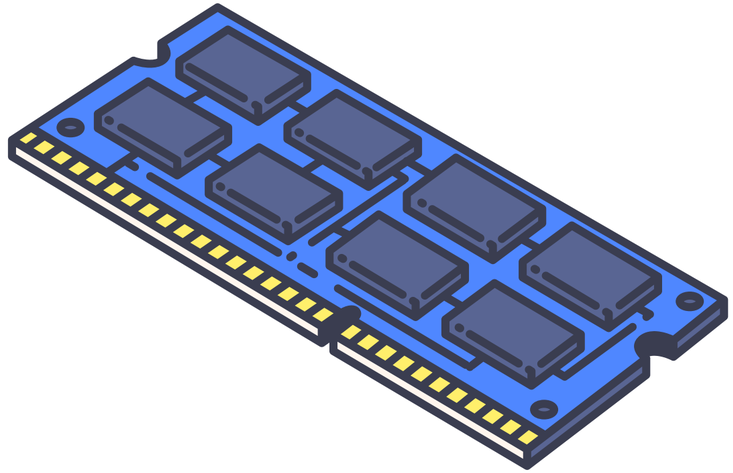
ENTENDENDO AS MEMÓRIAS

As memórias em um computador podem ser agrupadas pela pelo seu tipo e características. Basicamente temos os seguintes tipos: **registradores**, **cache**, **ROM**, **RAM**, **memória virtual** e memórias secundárias.

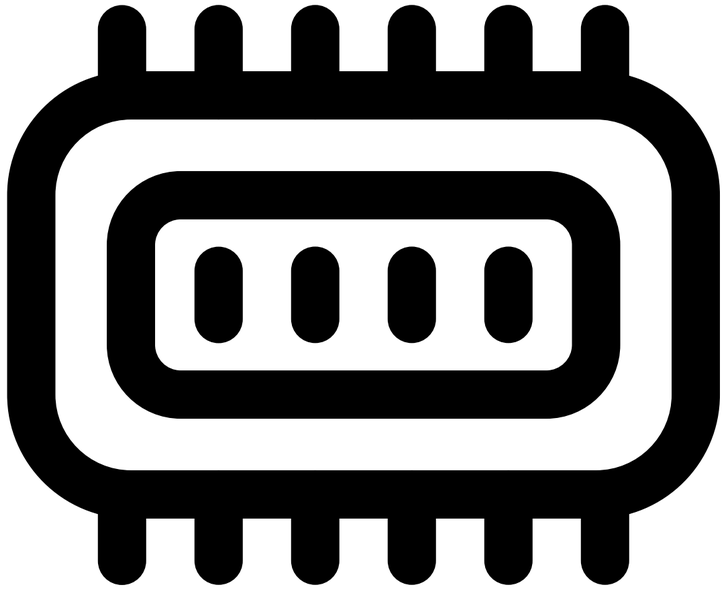


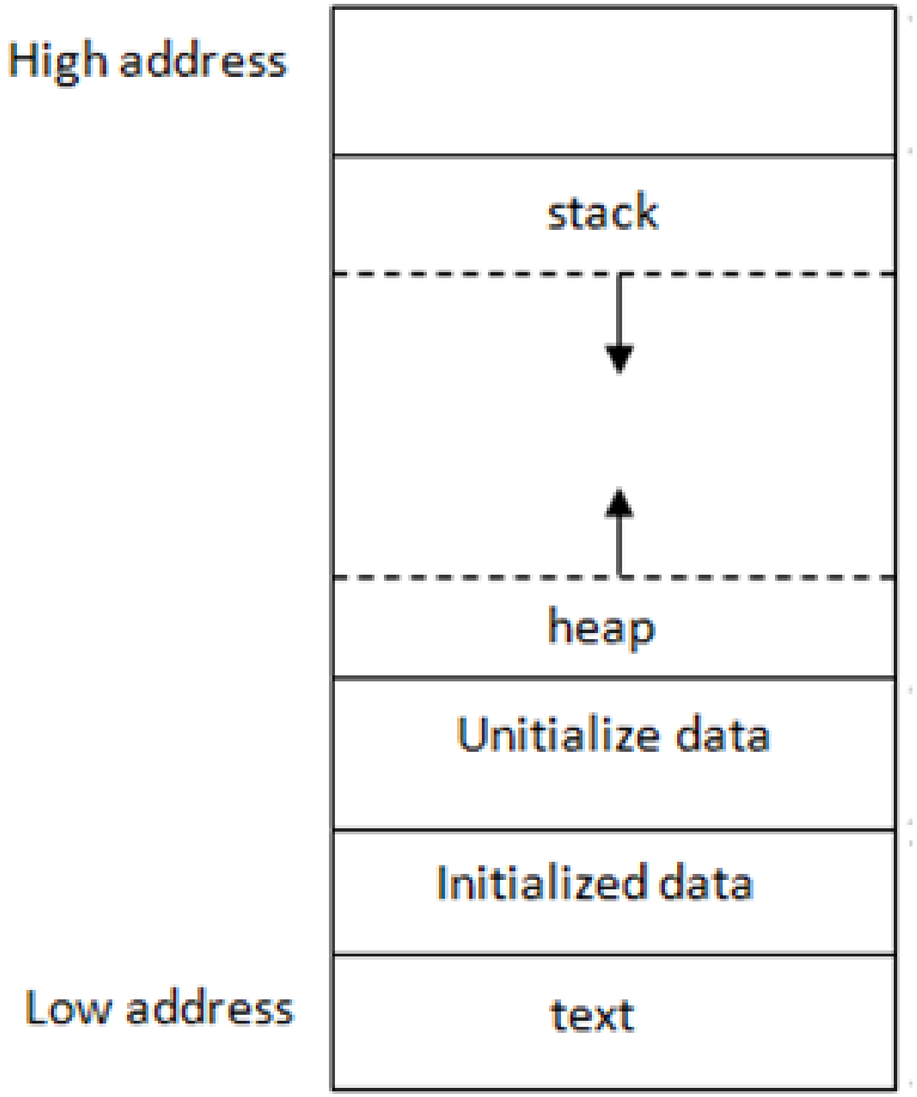
USO DA MEMÓRIA PELOS PROGRAMAS

Assim que um programa é inicializado o sistema operacional reserva uma quantidade de memória RAM para esse programa. Essa memória não é um bloco único de informações, mas um conjunto de áreas reservadas para fins específicos.



USO DA MEMÓRIA PELOS PROGRAMAS

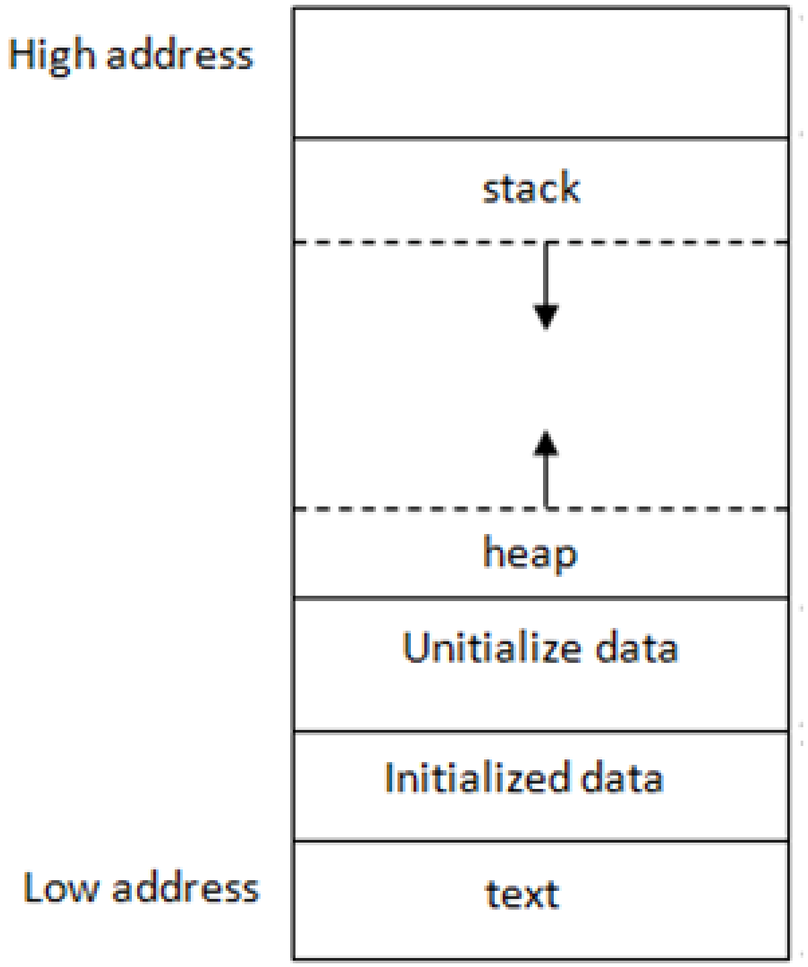




Memory layout of a c++ program

ELEMENTOS DA MEMÓRIA DE PROGRAMAS

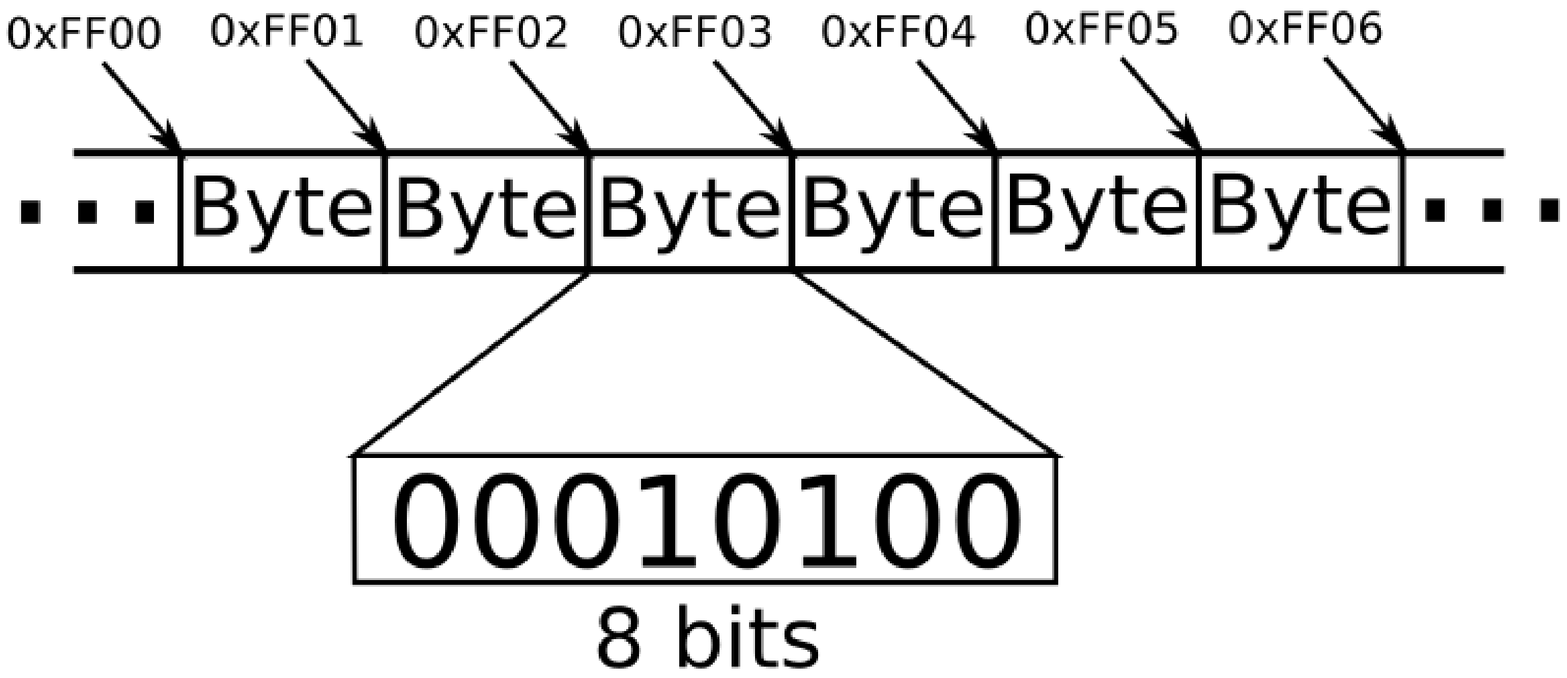
- stack: armazena variáveis locais
- heap: memória dinâmica para o programador alocar;
- data: armazena variáveis globais, separadas em inicializadas e não inicializadas;
- text: armazena o código do programa que está sendo executado.



Memory layout of a c++ program

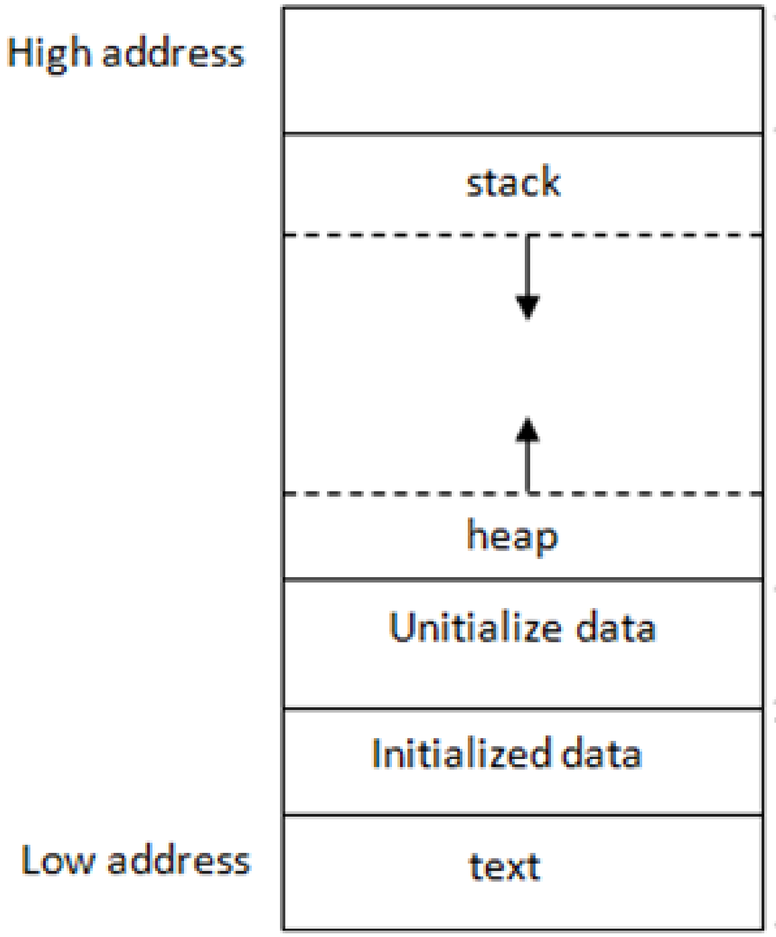
ELEMENTOS DA MEMÓRIA DE PROGRAMAS

Para identificar cada local de memória, atribuímos a cada byte de memória um “endereço”. Os endereços vão de 0 até o maior endereço possível, dependendo da máquina.



ELEMENTOS DA MEMÓRIA DE PROGRAMAS

Conforme a figura abaixo, os segmentos text, data, e heap possuem números de endereços baixos, enquanto a memória stack possui endereços maiores.



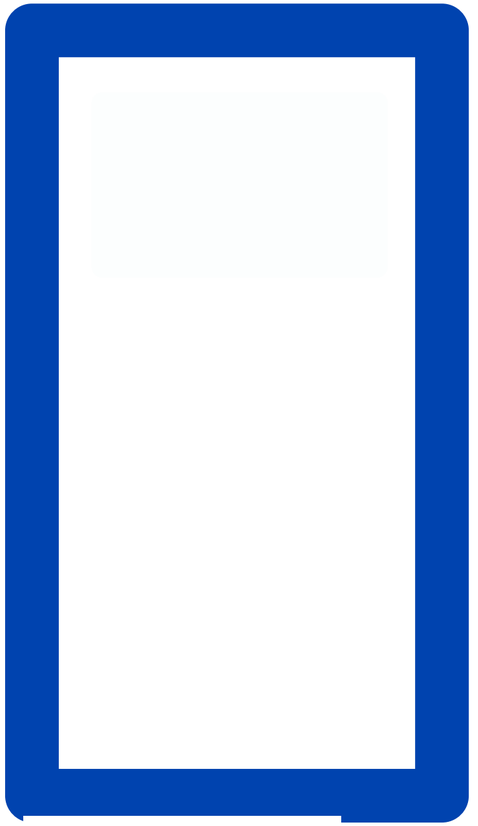
Memory layout of a c++ program

STACK (PILHA)

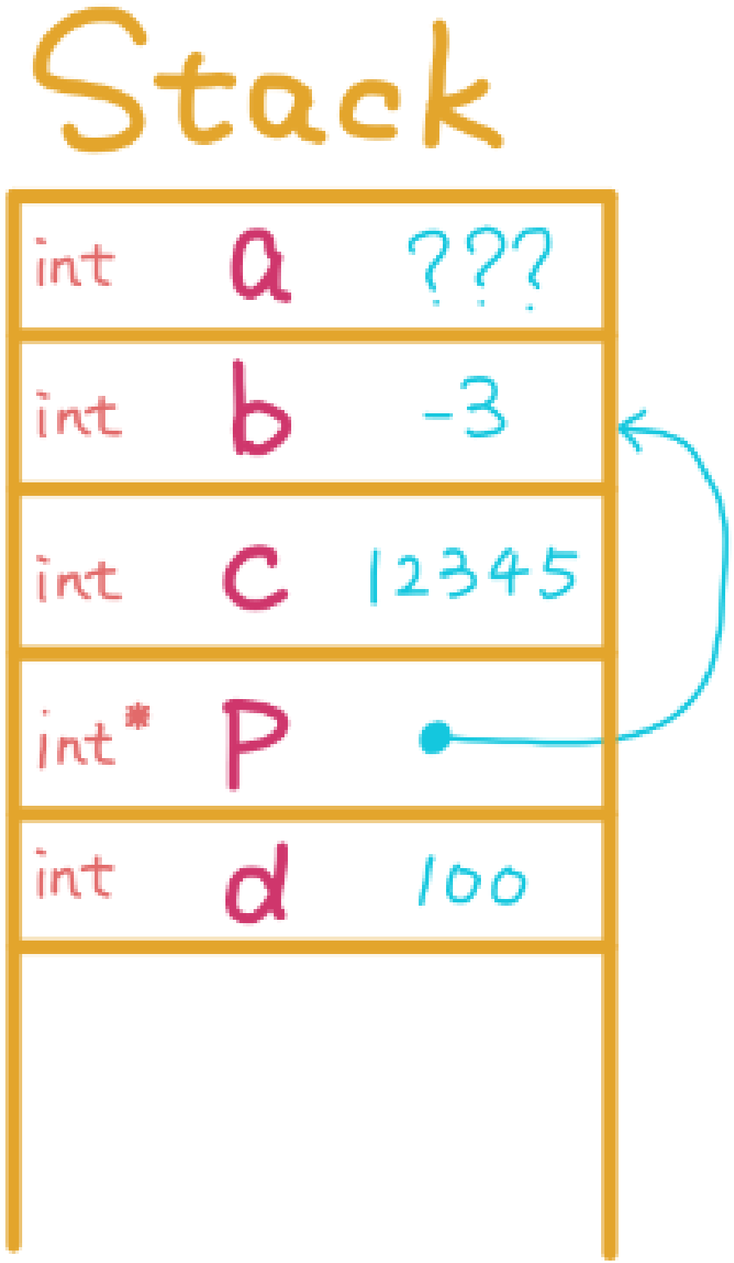
Refere-se ao segmento da memória que está próximo do topo da memória e que possui o endereço alto. Cada vez que uma função é chamada, a máquina aloca alguma memória na pilha para ela.

STACK (PILHA)

Tais alocações fazem a pilha crescer para baixo. Após o retorno da função, a memória da pilha dessa função é desalocada, o que significa que todas as variáveis locais se tornam inválidas. A alocação e desalocação da memória da pilha é feita automaticamente. As variáveis alocadas na pilha são chamadas de variáveis de pilha ou variáveis automáticas .



STACK (PILHA)



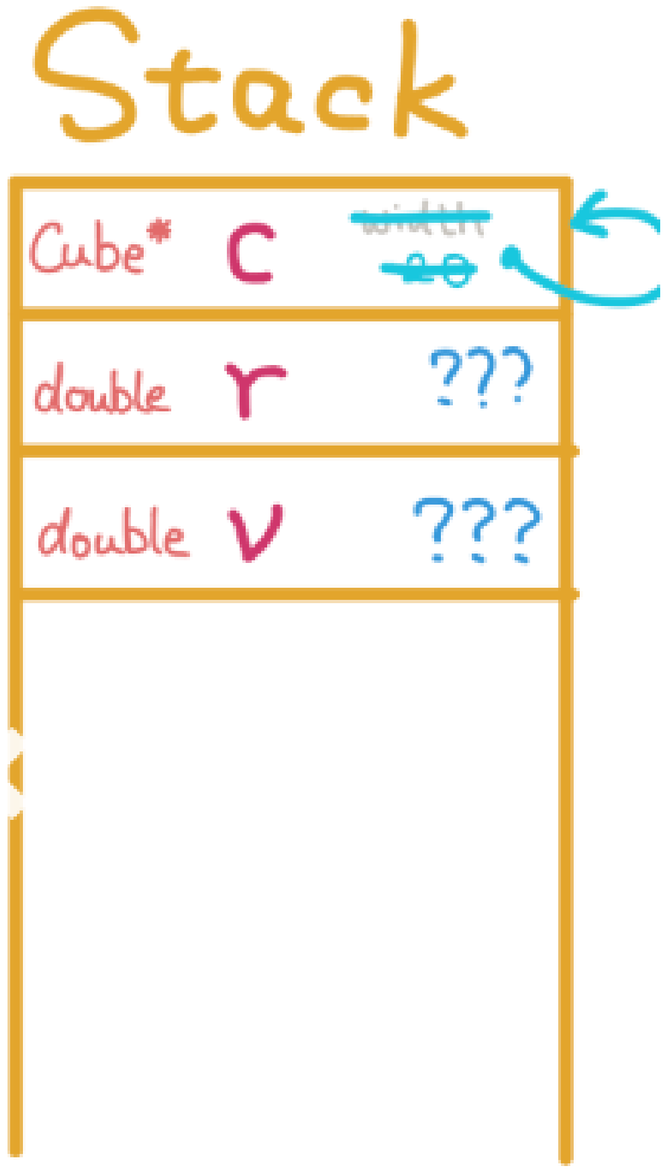

```c
int hello() {
  int a = 100;
  return a;
}
int main() {
  int a;
  int b = -3;
  int c = 12345;
  int *p = &b;
  int d = hello();
  return 0;
}
```

STACK (PILHA)

Como a memória da pilha de uma função é desalocada após o retorno da função, não há garantia de que o valor armazenado nessas áreas permanecerá o mesmo. Um erro comum é retornar um ponteiro para uma variável de pilha em uma função auxiliar. Depois que o chamador obtém esse ponteiro, a memória de pilha inválida pode ser substituída a qualquer momento.

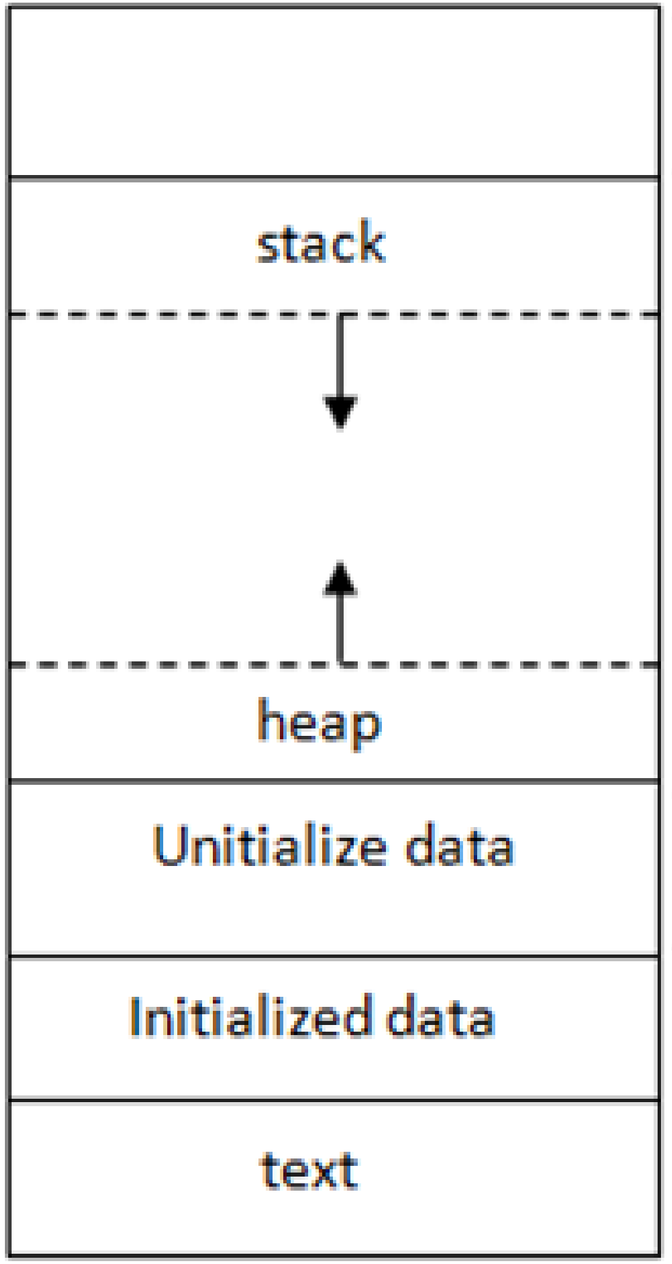
STACK (PILHA)

```
Cube *CreateCube() {
  Cube c(20);
  return &c;
}
int main() {
  Cube *c = CreateCube();
  double r = c->getVolume();
  double v = c->getSurfaceArea();
  return 0;
}
```



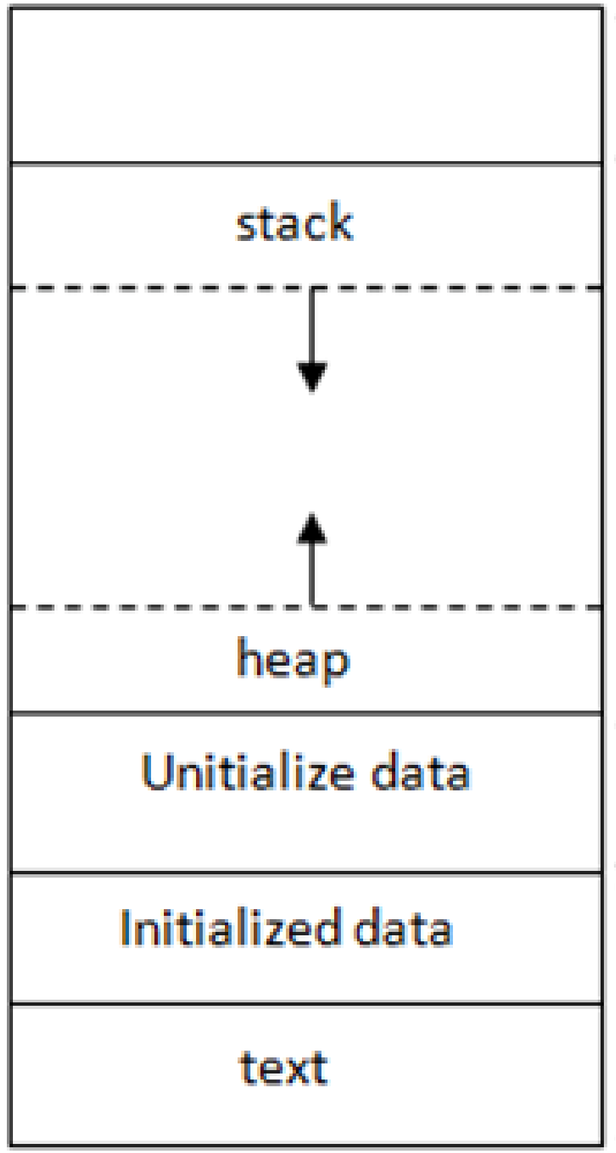
HEAP

As funções não podem retornar ponteiros de variáveis que estão na pilha. Para resolver esse problema, devemos retornar por cópia ou colocar os valor em algum lugar mais permanente do que a memória de pilha. A memória heap é um desses lugares.



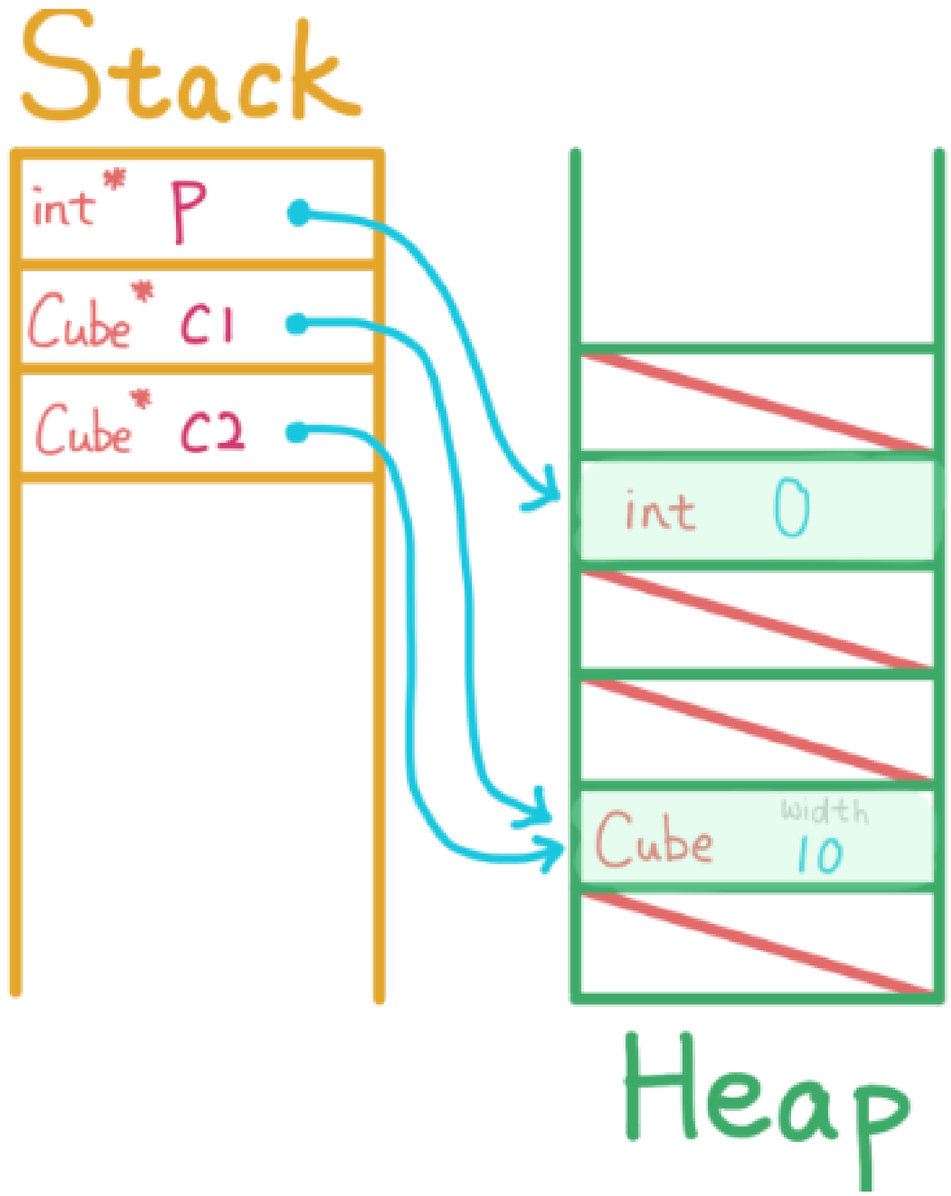
HEAP

Ao contrário da memória de pilha, a memória heap é alocada explicitamente pelos programadores e não será desalocada até que seja explicitamente liberada. Para alocar memória heap em C++, devemos usar a palavra-chave new seguida do construtor do que se deseja alocar. O valor de retorno do new será o endereço do que você acabou de criar (que aponta para algum lugar no heap).

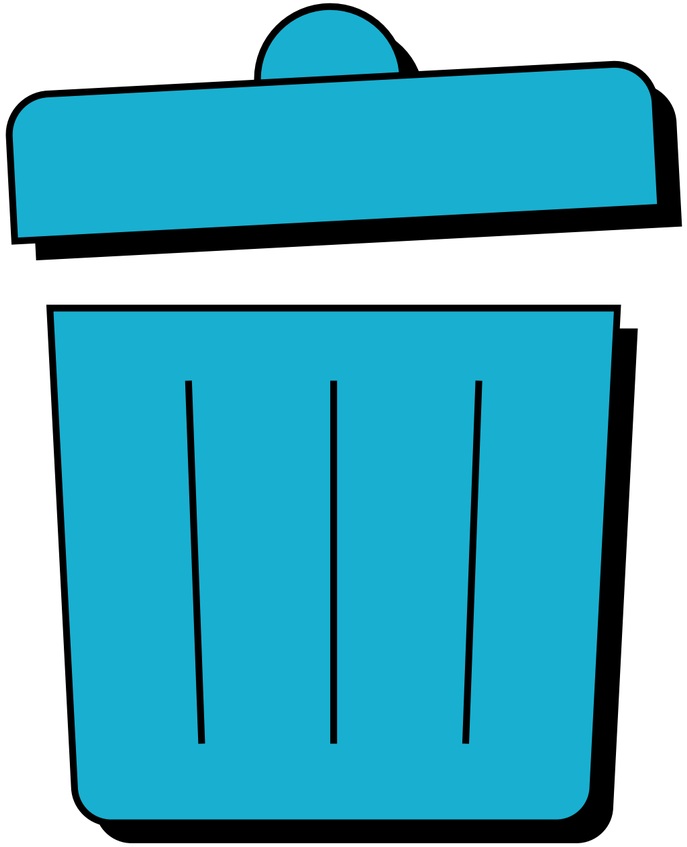


HEAP

```
int main() {
   int *p = new int;
   Cube *c1 = new Cube();
   Cube *c2 = c1;
   c2->setLength( 10 );
   return 0;
}
```



HEAP



Para liberar memória heap, use a palavra-chave delete seguida do ponteiro alocado para a memória heap. Tenha cuidado com a memória que você liberou. Se você tentar usar os ponteiros para essas memórias depois de liberá-los, isso causará um comportamento indefinido.

TIPOS DE DADOS

TIPOS DE DADOS - ESCALARES

Refere-se a forma mais simples de repreentar um dado no computador. Internamente são vistos como uma sequencia de bytes.

O formato de representação interna, ou seja, como uma sequência de bits é traduzida para um valor, pode variar de computador para computador, embora haja um esforço crescente para uniformizar a representação de tipos básicos

Dec	Hex	Oct	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr	Dec	Hex	Oct	HTML	Chr
0	0	000	NULL	32	20	040	&#032;	Space	64	40	100	&#064;	@	96	60	140	&#096;	`
1	1	001	Start of Header	33	21	041	&#033;	!	65	41	101	&#065;	A	97	61	141	&#097;	a
2	2	002	Start of Text	34	22	042	&#034;	"	66	42	102	&#066;	B	98	62	142	&#098;	b
3	3	003	End of Text	35	23	043	&#035;	#	67	43	103	&#067;	C	99	63	143	&#099;	c
4	4	004	End of Transmission	36	24	044	&#036;	$	68	44	104	&#068;	D	100	64	144	&#100;	d
5	5	005	Enquiry	37	25	045	&#037;	%	69	45	105	&#069;	E	101	65	145	&#101;	e
6	6	006	Acknowledgment	38	26	046	&#038;	&	70	46	106	&#070;	F	102	66	146	&#102;	f
7	7	007	Bell	39	27	047	&#039;	'	71	47	107	&#071;	G	103	67	147	&#103;	g
8	8	010	Backspace	40	28	050	&#040;	(	72	48	110	&#072;	H	104	68	150	&#104;	h
9	9	011	Horizontal Tab	41	29	051	&#041;	)	73	49	111	&#073;	I	105	69	151	&#105;	i
10	A	012	Line feed	42	2A	052	&#042;	*	74	4A	112	&#074;	J	106	6A	152	&#106;	j
11	B	013	Vertical Tab	43	2B	053	&#043;	+	75	4B	113	&#075;	K	107	6B	153	&#107;	k
12	C	014	Form feed	44	2C	054	&#044;	,	76	4C	114	&#076;	L	108	6C	154	&#108;	l
13	D	015	Carriage return	45	2D	055	&#045;	-	77	4D	115	&#077;	M	109	6D	155	&#109;	m
14	E	016	Shift Out	46	2E	056	&#046;	.	78	4E	116	&#078;	N	110	6E	156	&#110;	n
15	F	017	Shift In	47	2F	057	&#047;	/	79	4F	117	&#079;	O	111	6F	157	&#111;	o
16	10	020	Data Link Escape	48	30	060	&#048;	0	80	50	120	&#080;	P	112	70	160	&#112;	p
17	11	021	Device Control 1	49	31	061	&#049;	1	81	51	121	&#081;	Q	113	71	161	&#113;	q
18	12	022	Device Control 2	50	32	062	&#050;	2	82	52	122	&#082;	R	114	72	162	&#114;	r
19	13	023	Device Control 3	51	33	063	&#051;	3	83	53	123	&#083;	S	115	73	163	&#115;	s
20	14	024	Device Control 4	52	34	064	&#052;	4	84	54	124	&#084;	T	116	74	164	&#116;	t
21	15	025	Negative Ack.	53	35	065	&#053;	5	85	55	125	&#085;	U	117	75	165	&#117;	u
22	16	026	Synchronous idle	54	36	066	&#054;	6	86	56	126	&#086;	V	118	76	166	&#118;	v
23	17	027	End of Trans. Block	55	37	067	&#055;	7	87	57	127	&#087;	W	119	77	167	&#119;	w
24	18	030	Cancel	56	38	070	&#056;	8	88	58	130	&#088;	X	120	78	170	&#120;	x
25	19	031	End of Medium	57	39	071	&#057;	9	89	59	131	&#089;	Y	121	79	171	&#121;	y
26	1A	032	Substitute	58	3A	072	&#058;	:	90	5A	132	&#090;	Z	122	7A	172	&#122;	z
27	1B	033	Escape	59	3B	073	&#059;	;	91	5B	133	&#091;	[	123	7B	173	&#123;	{
28	1C	034	File Separator	60	3C	074	&#060;	<	92	5C	134	&#092;	\	124	7C	174	&#124;	|
29	1D	035	Group Separator	61	3D	075	&#061;	=	93	5D	135	&#093;	]	125	7D	175	&#125;	}
30	1E	036	Record Separator	62	3E	076	&#062;	>	94	5E	136	&#094;	^	126	7E	176	&#126;	~
31	1F	037	Unit Separator	63	3F	077	&#063;	?	95	5F	137	&#095;	_	127	7F	177	&#127;	Del

asciicharstable.com

TABELA ASCII

TIPOS DE DADOS - OS ESCALARES

Em geral a linguagem de programação definem tipos para seus escalares, (ex: int, float, double, char, boolean e suas subdivisões).

Tipo	Tamanho (em bits)	Intervalo
Char	8	-128 a 127
Int	16	-32768 a 32767
Float	32	3,4E-38 a 3,4E+38
double	64	1,7E-308 a 1,7E+308
void	0	sem valor

TIPOS DE DADOS - PONTEIROS

Ponteiros são variáveis que armazenam endereços de memória de outras variáveis. Eles permitem o acesso e a manipulação dessas variáveis indiretamente, referenciando-as por meio do endereço de memória que elas ocupam. Em outras palavras, um ponteiro "aponta" para uma posição de memória que contém um valor específico.

TIPOS DE DADOS - COMPOSTOS

Os tipos de dados compostos são tipos de dados que agrupam valores simples de outros tipos de dados. Eles são úteis para armazenar informações mais complexas em uma única variável.

Alguns exemplos de tipos de dados compostos incluem:

- Listas, conjuntos,
- Arrays (vetores e matrizes)
- Dicionarios, tuplas
- Structs, enumerações,

QUESTÃO 01 - BANCA CESPE

Durante a execução de um processo computacional na plataforma Linux ou Windows, é possível identificar a presença de várias áreas de memória distintas, entre elas: área de pilha (Stack), área de alocação dinâmica de memória (Heap), área de dados estáticos (Data) e área de código (Code).

- Certo
- Errado

Ano: 2015 Banca: CESPE / CEBRASPE Órgão: FUB Prova: CESPE - 2015 - FUB - Técnico de Tecnologia da Informação

DÚVIDAS???