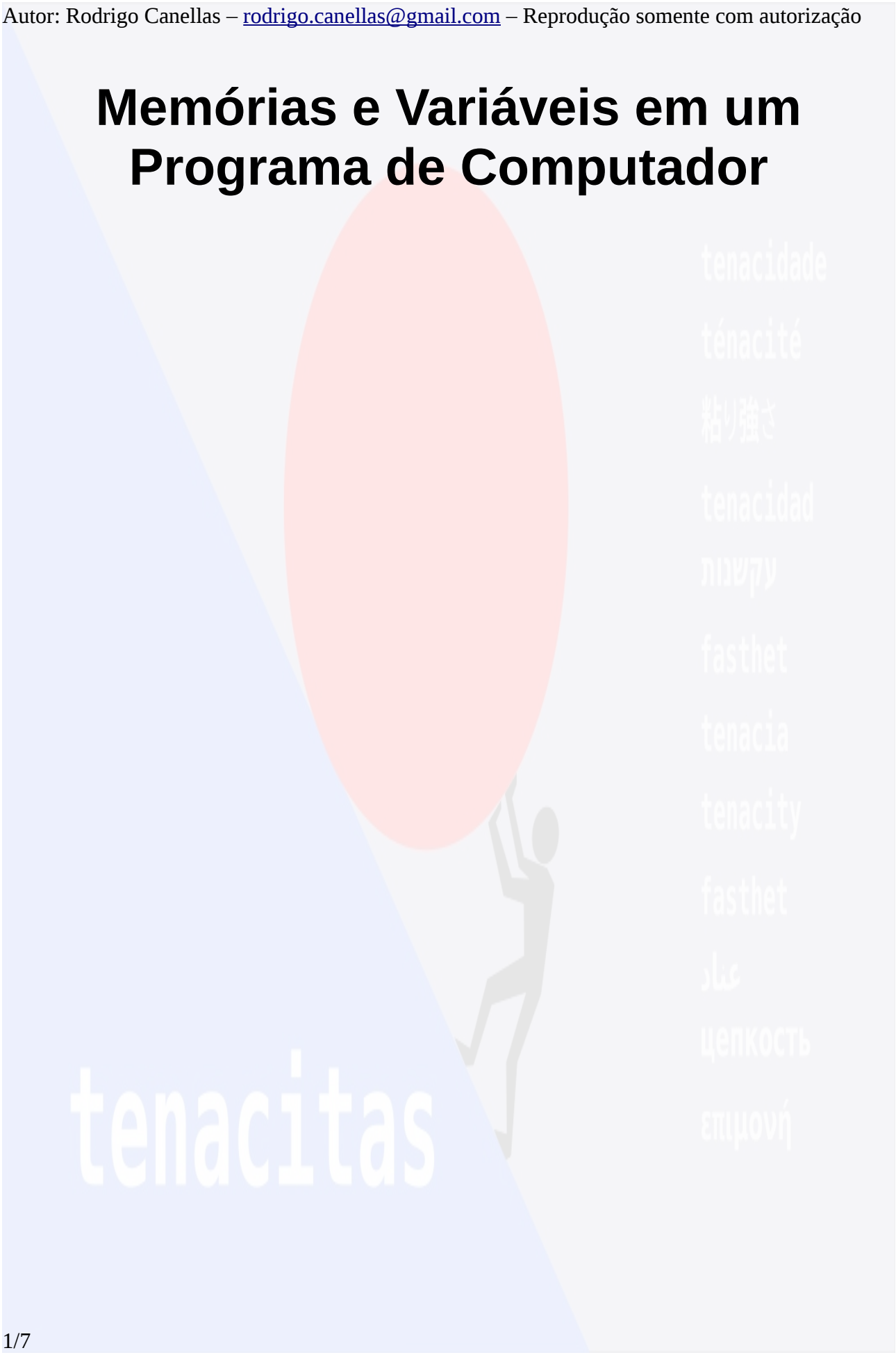
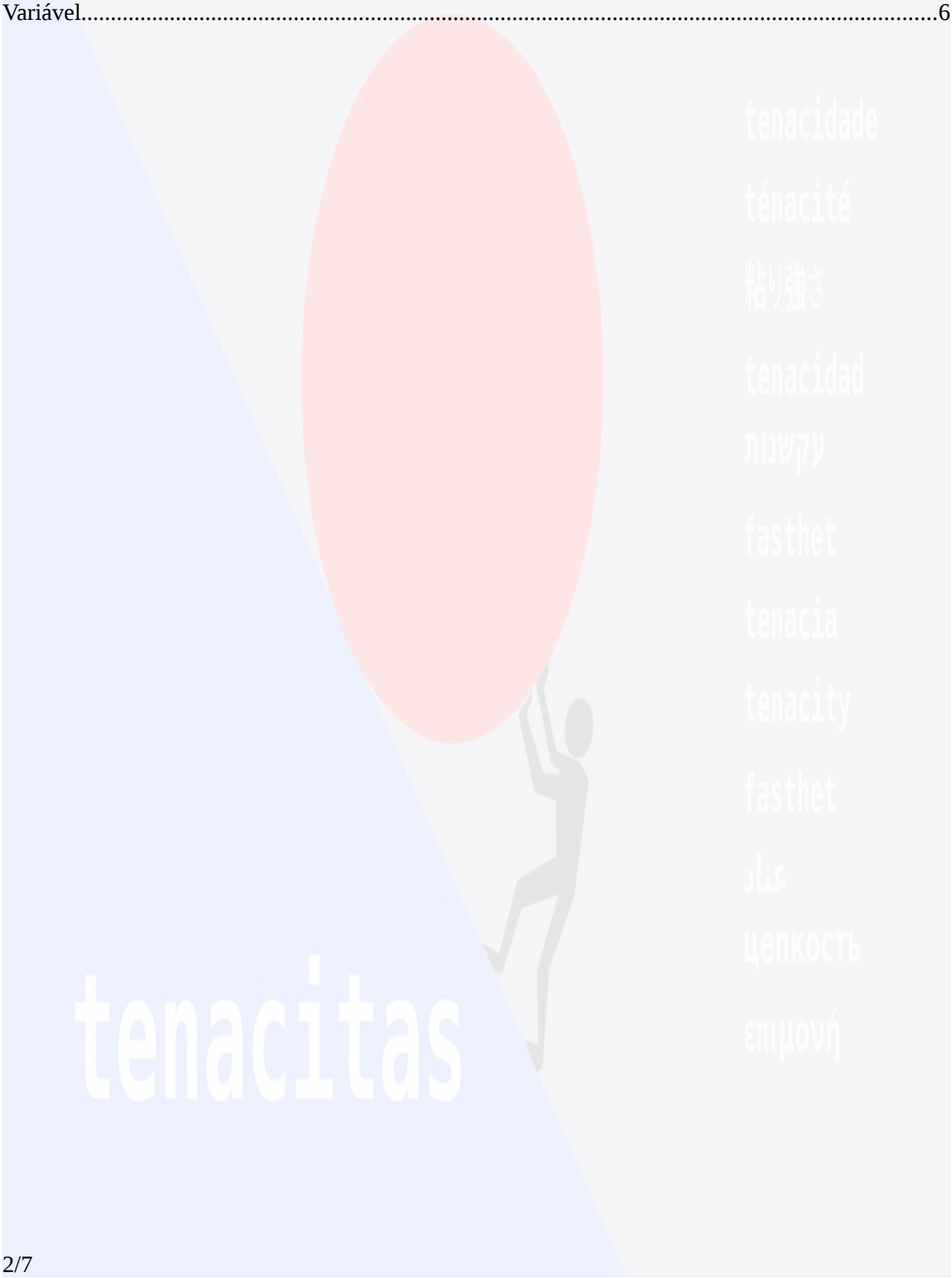
Memórias e Variáveis em um Programa de Computador



Sumário



Áreas de memória

Definimos como um programa de computador um arquivo executável. Um programa é identificado pelo nome deste arquivo, e está localizado em um diretório em uma memória de armazenamento. Um programa tem duas áreas de memória, geradas no processo de construção (compilação e ligação): CODE e STATIC. Na área CODE está o código do programa, i.e., as funções que serão executadas. Na área STATIC estão as variáveis declaradas e definidas em tempo de compilação, e cujos ciclos de vida coincidem com o da execução do programa.

STATIC	
CODE	

Definimos como um processo, um programa que está sendo executado. Um processo é identificado por um identificador, atribuído pelo Sistema Operacional, e comumente chamado de PID. Um processo tem, além das áreas de memória STATIC e CODE, as áreas de memória STACK e HEAP. Na STACK ficam os Registros de Ativação de cada função, que são áreas de memória dentro da STACK onde as variáveis locais e os argumentos de uma função são criados. Na HEAP ficam as variáveis que são dinamicamente criadas.

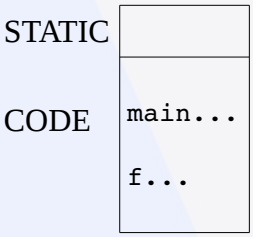
HEAP	
STACK	
STATIC	
CODE	

Por exemplo, no código abaixo:

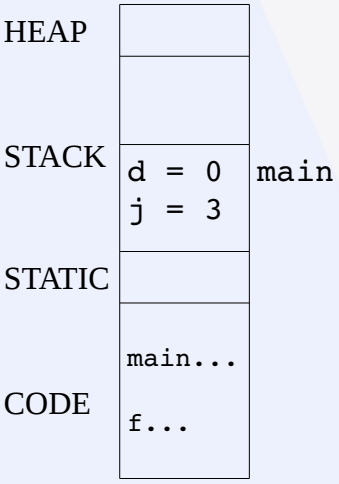
```
01 #include <stdio.h>
02
03 double f(int i);
04
05 int main() {
06     double d = 0.0;
07     int j = 3;
08     d = f(j);
09     printf("i = %d, d = %lf\n", i, d);
10     return 0;
11 }
12
13 double f(int i) {
14     return 2.5 * i;
```

```
15 }
```

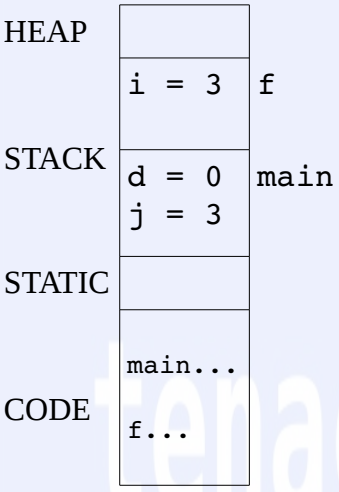
Após a geração do programa, teríamos:



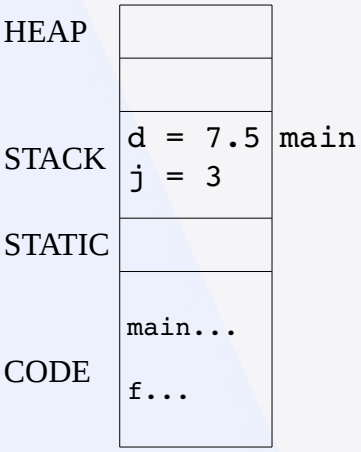
Ao ser iniciado, a função `main` é a primeira a ser executada, e ganha uma área de memória na STACK, seu registro de ativação, para as variáveis locais e argumentos:



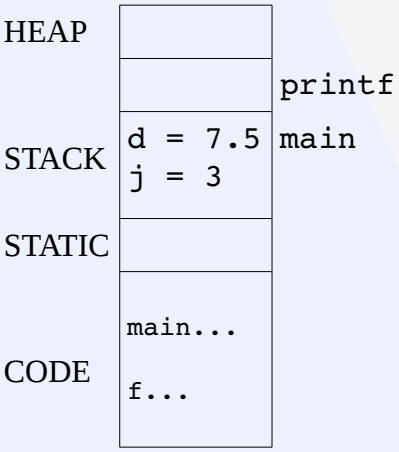
Na linha 8, a função `f` é chamada:



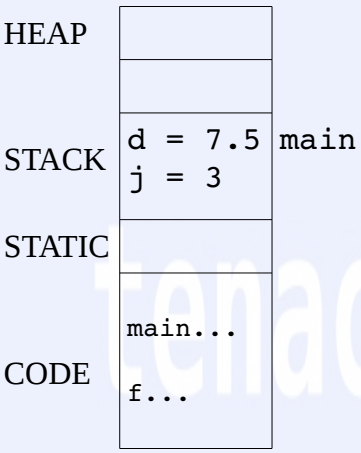
Quando `f` acaba sua execução, seu registro de ativação é desempilhado:



Na linha 9, `printf` é chamada:



Quando `printf` acaba, seu registro de ativação é desempilhado:



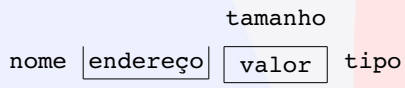
Quando `main` acaba, o processo é retirado de memória.

Variável

Uma variável é uma área de memória de um programa, ou processo, com as seguintes características:

- nome
- tipo
- tamanho
- valor
- endereço

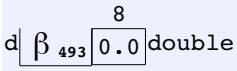
Usaremos a seguinte representação gráfica de uma variável:



Para mais facilmente identificar a qual a área de memória um endereço se refere, usaremos a seguinte convenção:

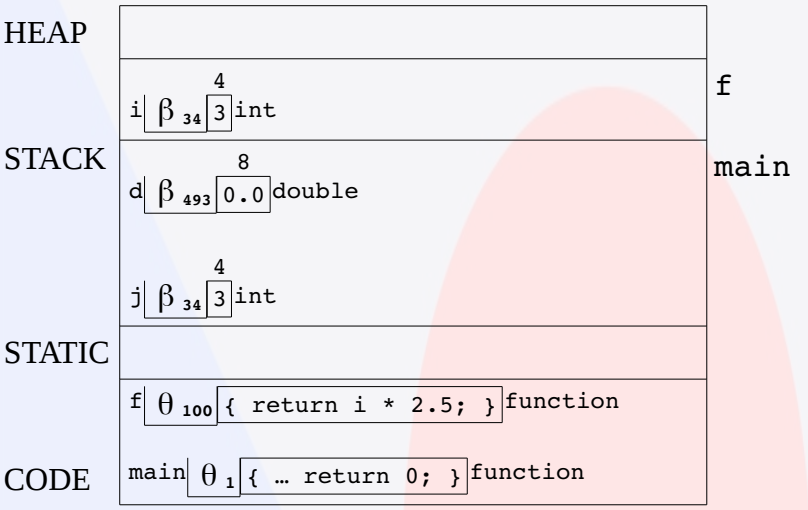
CODE	θ
STATIC	α
STACK	β
HEAP	ρ

Assim, a variável d, na linha 6 do programa anterior seria representada



O endereço `493` é um número qualquer, não há uma correspondência fidedigna com qualquer endereço físico.

Redesenhando as áreas de memória do processo quando a função `f` é chamada, na linha 8:



Obs: retiramos da descrição de `function` o tamanho porque não há como determinar o tamanho de uma função