MC-102 — Aula 03
Comandos de Entrada e Saída

Instituto de Computação – Unicamp

Segundo Semestre de 2009

Roteiro

- Saídade dados
- Entrada de dados

Exercício sobre tipos

```
int main() {
   int a;
   unsigned int b;
   short int c;
   unsigned short int d;
   float e;
   double f;
   char g;
   unsigned char h;

   a = 10;
   b = -6;
   c = 100000;
   d = 33000;
   e = -80000.657;
   f = 30;
   g = 'a';
   g = a;
   h = 200;
   h = 'B';
}
```


Um parêntese: comentários

- O código fonte pode conter comentários direcionados unicamente ao programador. Estes comentários devem estar delimitados pelos símbolos /* e */, e são ignorados pelo compilador.

Exemplo

```
#include <stdio.h>

/* Este é o meu primeiro programa. */
main() {
  printf("Hello, world!\n");
}
```

- Comentários são úteis para descrever o algoritmo usado e para explicitar suposições não óbvias sobre a implementação.

Escrevendo o conteúdo de uma variável na tela

- Podemos imprimir, além de texto puro, o conteúdo de uma variável utilizando o comando `printf`. Para isso, utilizamos um símbolo no texto para representar que aquele trecho deve ser substituído por uma variável e, no final, passamos uma lista de variáveis ou constantes, separadas por vírgula.

Exemplo

```
printf("A variável %s contém o valor %d","a", a);
```
imprime `A variável a contém o valor 10`

- Nesse caso, %s deve ser substituído por uma variável ou constante do tipo `string` enquanto %d deve ser substituído por uma variável ou constante do tipo inteiro.

Formatos inteiros

%d — Escreve um inteiro na tela sem formatação.

Exemplo

```
printf ("%d", 10);
```
imprime 10

Formatos inteiros

%< *numero* >d — Escreve um inteiro na tela, preenchendo com espaços a esquerda para que ele ocupe pelo menos < *numero* > casas na tela.

Exemplo

```
printf ("%4d", 10);
```
imprime < *espaco* >< *espaco* >10

Formatos inteiros

%0< *numero* >d — Escreve um inteiro na tela, preenchendo com zeros a esquerda para que ele ocupe pelo menos comprimento < *numero* >.

Exemplo

```
printf ("%04d", 10);
```
imprime 0010

Formatos inteiros

%< $numero_1$ >.0< $numero_2$ >d — Escreve um inteiro na tela, preenchendo com espaços a esquerda para que ele ocupe pelo menos < $numero_1$ > casas na tela e com zeros para que ele possua pelo menos comprimento < $numero_2$ >.

Exemplo

```
printf ("%6.04d", 10);
```

imprime < *espaco* >< *espaco* >0010

Formatos inteiros

- A letra d pode ser substituida pelas letras u e l, ou as duas, quando desejamos escrever variáveis do tipo unsigned ou long, respectivamente.

Exemplo

```
printf ("%d", 4000000000);
```

escreve -294967296 na tela, enquanto que

```
printf ("%u", 4000000000);
```

escreve 4000000000.

Formatos ponto flutuante

%f — Escreve um ponto flutuante na tela, sem formatação

Exemplo

```
printf ("%f", 10.0);
```

imprime 10.000000

Formatos ponto flutuante

%e — Escreve um ponto flutuante na tela, em notação científica

Exemplo

```
printf ("%e", 10.02545);
```

imprime 1.002545e+01

Formatos ponto flutuante

%< *tamanho* >.< *decimais* >f — Escreve um ponto flutuante na tela, com tamanho < *tamanho* > e < *decimais* > casas decimais. Lembre-se que o ponto, utilizado para separar a parte inteira da decimal, também conta no tamanho.

Exemplo

```
printf ("%6.2f", 10.0);
```
imprime < *espaco* >10.00

Formatos ponto flutuante

- A letra f pode ser substituida pelas letras lf, para escrever um `double` ao invés de um `float`

Exemplo

```
printf ("%6.2lf", 10.0);
```
imprime < *espaco* >10.00

Formato caracter

%c — Escreve uma letra.

Exemplo

```
printf ("%c", 'A');
```
imprime a

Note que `printf ("%c", 65)` também imprime a letra A.

Formato `string`

%s — Escreve uma string

Exemplo

```
printf ("%s", "Meu primeiro programa");
imprime Meu primeiro programa
```

Exemplo 1

```
printf ("%s\n", "Meu primeiro programa");
imprime Meu primeiro programa <nova linha>
```

Exemplo 2

```
printf ("%s\t%s", "Meu primeiro","programa");
imprime Meu primeiro <tabulação> programa
```

A função scanf

- Realiza a leitura de um texto a partir do teclado.
- Parâmetros:
  - Uma string, indicando os tipos das variáveis que serão lidas e o formato dessa leitura.
  - Uma lista de variáveis.
- Aguarda que o usuário digite um valor e atribui o valor digitado à variável.

A função scanf

O programa abaixo é composto de quatro passos:

1. Cria uma variável `n`;
2. Escreve na tela `Digite um número:`
3. Lê o valor do número digitado
4. Imprime o valor do número digitado

```
#include <stdio.h>
main(){
  int n;
  printf("Digite um número: ");
  scanf("%d",&n);
  printf("O valor digitado foi %d\n",n);
}
```


A função scanf

Leitura de várias variáveis:

```
#include <stdio.h>
main(){
  int m, n, o;
  printf("Digite três números: ");
  scanf("%d %d %d",&m, &n, &o);
  printf("O valores digitados foram\
          %d %d %d\n", m, n, o);
}
```


O operador “address-of” & de C

Toda variável tem um endereço de memória associado a ela. Esse endereço é o local onde essa variável é armazenada no sistema. O operador & retorna o endereço de uma determinada variável

Exemplo

`printf ("%d", &valor);`
Imprime o endereço da variável `valor`.

O operador "address-of" & de C

- É necessário usar o operador & no comando `scanf`, pois esse operador indica que o valor digitado deve ser colocado no endereço referente a uma variável.
- **Esquecer de colocar o & comercial é um erro muito comum que pode ocasionar erros de execução**.

O operador "address-of" & de C

O programa abaixo imprime o valor e o endereço da variável:

```
#include <stdio.h>
int main(void){
  int n = 8;
  printf("valor %d, endereço 0x%x\n",n,&n);
}
```


Formatos de leitura de variável

Os formatos de leitura são muito semelhantes aos formatos de escrita utilizados pelo `printf`. A tabela a seguir mostra alguns formatos possíveis de leitura

Código	Função
%c	Lê um único caracter
%s	Lê uma série de caracteres

Formatos de leitura de variável

Código	Função
%d	Lê um número decimal
%u	Lê um decimal sem sinal
%l	Lê um inteiro longo
%f	Lê um número em ponto flutuante
%lf	Lê um double