### SCC-211 Lab. Algoritmos Avançados Capítulo 1

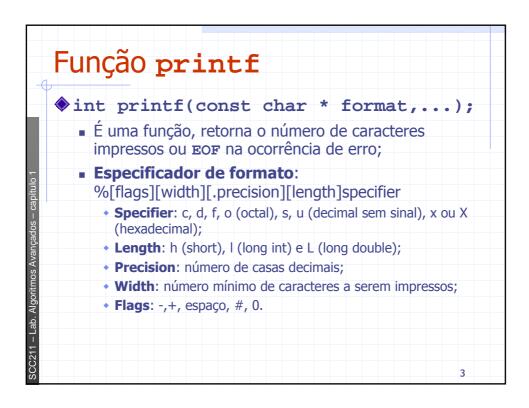
Entrada e Saída

João Luís G. Rosa

....

### Principais Funções

- \*#include<stdio.h>
  - printf impressão formatada em stdout;
  - sprintf impressão formata em strings;
  - gets leitura de strings de stdin
    (depreciado);
  - fgets leitura de strings de streams;
  - scanf leitura formatada de stdin;
  - sscanf leitura formatada de strings;
  - getchar leitura de caractere de stdin.



```
Finclude<stdio.h>
$include<stdib.h>
int main()
{

printf("Caracteres: %c %c, ASCII: %d\n", 'a', 65, '2');
printf("Decimais: %d %ld\n", 1977, 650000);
printf("Precedido com brancos: %10d \n", 1977);
printf("Precedido com zeros: %010d \n", 1977);
printf("Em decimal: %d, octal: %o e hexadecimal %fx\n", 100, 100, 100);
printf("Truque com a largura: %*d\n", 5, 10);
printf("%s\n", "Uma string");
system("pause");
return 0;
}
```

# Saídas Caracteres: a A, ASCII: 90 Decimais: 1977 650000 Precedido com brancos: 1977 Precedido com zeros: 0000001977 Em decimal: 100, octal: 144 e hexadecimal 0x64 Ponto flutuante: 3.14 +3e+000 3.141600E+000 Truque com a largura: 10 Uma string

### Função gets

- \$char \* gets ( char \* str );
  - Depreciado por não permitir especificar o tamanho da string;
  - Realiza a leitura de caracteres até encontrar um caractere de nova linha ('\n') ou fim de arquivo;
  - Remove o caractere '\n' de stdin, mas não o coloca em str;
  - Insere o caractere `\o' no final de str.

### Função fgets

- char \* fgets ( char \* str, int
  num, FILE \* stream );
  - Realiza a leitura de caracteres até num-1
     caracteres ou encontrar um caractere de nova
     linha ou fim de arquivo;
  - O caractere '\n' é considerado válido e é inserido em str (permite identificar se ainda há algo no buffer de entrada - diferente de gets!);
  - Insere o caractere '\0' no final de str.

```
Função fgets

#include<stdio.h>
#include<stdlib.h>
int main() {
    char str[20];

    gets(str);
    printf("-%s-\n", str);
    fgets(str, 20, stdin);
    printf("-%s-\n", str);
    system("pause");
    return 0;
}
```

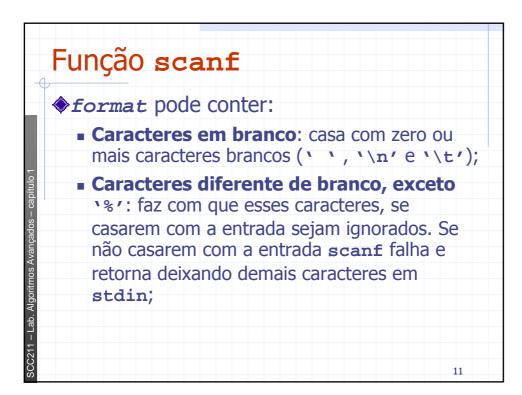
### Retorno: gets e fgets

- ◆Para ambos gets e fgets:
  - Em caso de sucesso, as funções retornam o parâmetro str;
  - Se o caractere de fim de arquivo é encontrado e nenhum caractere foi lido, então um ponteiro NULL é retornado;
  - Se um erro é encontrado NULL é retornado;
  - ferror() e feof() podem ser utilizadas para diferenciar entre erros e fim de arquivo.

9

### Função scanf

- int scanf(const char \* format,
  ...);
- **♦** format pode conter:
  - Especificador de formato: %[\*][width][modifiers]type
    - **type**: c, d, f, o (octal), s, u, x, X (hexa), n (nr. de valores lidos);
    - modifiers: h (short), I (long) e L (long double);
    - width: especifica o número máximo de caracteres;
    - \*: faz com que os dados sejam lidos de stdin, mas ignorados.



```
Exemplo 1: scanf
   #include<stdio.h>
   #include<stdlib.h>
   int main() {
      int n, m;
       char s[10];
       scanf("% *d %d", &n);
       printf("Valor lido: %d\n", n);
       scanf("%7d%s", &n, s);
      printf("Valor lido: %d, %s\n", n, s);
       scanf("%d %d", &n, &m);
      printf("Valor lido: %d, %d\n", n, m);
       scanf("%d.%d", &n, &m);
       printf("Parte inteira %d, parte fracionaria %d\n", n, m);
       system("pause");
       return 0;
                                                                      12
```

```
Saídas

123
3
Valor lido: 3
12345678
Valor lido: 1234567, 8
123 34
Valor lido: 123, 34
2.3
Parte inteira 2, parte fracionaria 3
```

```
#include<stdio.h>
        #include<stdlib.h>
        int main() {
             int n, m;
             char s[10];
             char c;
            scanf("%d", &n);
scanf("%c", &c);
             printf("Valor lido: %d, %c\n", n, c);
SCC211 - Lab. Algoritmos Avançados - capítulo
             scanf("%s", &s);
             c = getchar();
             printf("Valor lido: %s, %c\n", s, c);
             scanf("%d\n", &n);
             scanf("%c", &c);
             printf("Valor lido: %d, %c\n", n, c);
             scanf("%s ", &s);
             c = getchar();
             printf("Valor lido: %s, %c\n", s, c);
             system("pause");
             return 0;
                                                                                               14
```

### Função scanf A máscara %[] realiza a leitura de um string, cujos caracteres válidos são especificados entre os colchetes: scanf("%[abc]", s); //aabbccabc válido O símbolo ^ indica o conjunto complementar (qualquer caractere menos os presentes na lista): scanf("%[^abc], s); // defghijklmn... válido

### Função scanf

- Uma coisa importante sobre o scanf é o parâmetro de retorno:
  - Em caso de sucesso, mesmo que parcial,
     scanf retorna o número de itens lidos com sucesso.
    - Esse número pode ser um valor menor ou igual ao número de leituras esperado.
  - Em caso de falha antes de que qualquer dado seja lido com sucesso, a constante **EOF** é retornada.

17

### scanf("%s") VS. gets(fgets)

- \$scanf("%s") opera de forma diferente
  do gets(fgets):
  - Para o scanf, "%s" significa uma seqüência de caracteres diferente dos caracteres brancos. Portanto um scanf("%s") pode ler somente uma palavra de uma frase;
  - No gets e fgets, a linha toda é lida.

SCC211 - Lab Algoritmos Avanca



- Cuidado com fflush(stdin), pois não funciona em todos os compiladores!
  - "fflush is defined only for output streams. Since its definition of "flush" is to complete the writing of buffered characters (not to discard them), discarding unread input would not be an analogous meaning for fflush on input streams."

19

### **Sample input**int main() { int int i, int f;

1 10 100 200 201 210

900 1000

```
while (scanf("%d %d", &i, &j) != EOF) {
    // processa o caso de teste
}

int main() {
    int i, int f;

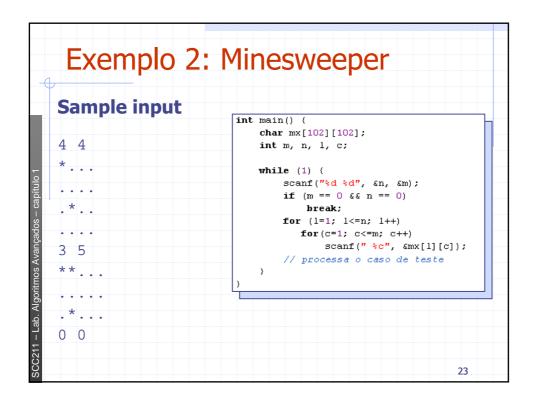
while (scanf("%d %d", &i, &j) == 2) {
        // processa o caso de teste
}
```

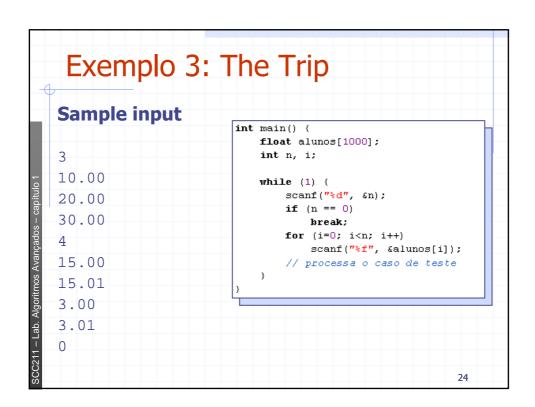
```
#include <stdio.h>

int main() {
    unsigned int i, j, k, aux, comp;
    int ciclo, maior_ciclo;

    printf("Entre com os valores de i e j: ");
    while (scanf("%u %u", &i, &j) != EOF)
    {
        printf("%u %u", i, j);

        if (i > j)
        {
            aux = i;
            i = j;
            j = aux;
        }
        21
```





```
Exemplo 4: Crypt Kicker Sample input
  and
  jane
  puff
  spot
  xsb qymm xsb rquat
                  scanf("%d\n", &dict_size);
  xxx yyyy zzz
                  for (i = 0; i < dict size; i++)
                      scanf("%s\n", words[i]);
                  while (gets(line)) {
                        line words last = 0;
                        pos = 0;
                       while (sscanf(&line[pos], "%s%n",
                             line_words[line_words_last++], &inc) != EOF)
                                   pos += inc;
                        //processa o caso de teste
```

# Referências Batista, G. & Campello, R. Slides disciplina Algoritmos Avançados, ICMC-USP, 2007. Skiena, S. S. & Revilla, M. A. Programming Challenges – The Programming Contest Training Manual. Springer, 2003.