Introdução à Computação
Repetições

Prof. Dr. Marcos Paulino Roriz Junior (marcosroriz@ufg.br)

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

ENGENHARIA DE TRANSPORTES

FCT
FACULDADE DE CIÊNCIAS E TECNOLOGIA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Repetições

Como calcular a média da sala?

- Instruções fluem em um única direção. Exemplo (média):

```
1.  p1 <- leia
2.  p2 <- leia
3.  média <- (p1 + p2) / 2
4.  SE média ≥ 6
5.  ENTÃO INÍCIO
6.          ESCREVA “aprovado(a)”
7.          FIM
8.  SENÃO INÍCIO
9.          ESCREVA “reprovado”
10.         FIM
11.ESCREVA media
```

```c
#include <stdio.h>

int main() {
    float p1, p2, media;
    scanf("%f", &p1);
    scanf("%f", &p2);
    media = (p1 + p2) / 2;
    if (media >= 6) {
        printf("aprovado(a)");
    }
    else {
        printf("reprovado(a)");
    }
    printf("%f", media);
}
```

Repetições

- Repetições permitem repetir uma determinada porção de instruções até uma condição seja satisfeita.

```
1. n <- 0
2. ENQUANTO n < 30 FAÇA
3.    INÍCIO
4.    p1 <- leia
5.    p2 <- leia
6.    média <- (p1 + p2) / 2
7.    SE média ≥ 6
8.    ENTÃO INÍCIO
9.          ESCREVA “aprovado(a)”
10.         FIM
11.   SENÃO INÍCIO
12.         ESCREVA “reprovado”
13.       FIM
14.   ESCREVA media
15.   n <- n + 1
16.   FIM
17.ESCREVA N
```

Repetições

- Repetições permitem repetir uma determinada porção de instruções até uma condição seja satisfeita.

```
1.  n <- 0
2.  ENQUANTO n < 30 FAÇA
3.     INÍCIO
4.     p1 <- leia
5.     p2 <- leia
6.     média <- (p1 + p2) / 2
7.     SE média ≥ 6
8.     ENTÃO INÍCIO
9.           ESCREVA "aprovado(a)"
10.          FIM
11.    SENÃO INÍCIO
12.          ESCREVA "reprovado"
13.       FIM
14.    ESCREVA media
15.    n <- n + 1
16.    FIM
17. ESCREVA N
```

A ausência desta instrução faz com o que laço se torne infinito

Repetições

1. ENQUANTO CONDIÇÃO FAÇA
2. INÍCIO
3. passo 1;
4. passo 2;
5. passo 3;
6. FIM

= bloco de instruções

- INSTRUÇÃO APÓS TÉRMINO DO ENQUANTO

Repetições

- Obter a média de temperatura de 5 sensores

```
1.  DECLARE M, V, N NUMÉRICO
2.  N <- 0
3.  M <- 0
4.  ENQUANTO N <= 10 FAÇA
5.  INÍCIO
6.      ESCREVA “DIGITE O VALOR”
7.      LEIA V
8.      M <- M + V
9.      N <- N + 1
10. FIM
11. M <- M / 10
12. ESCREVA M
```

Repetições

- Obter a média de temperatura de 5 sensores

```
1.  DECLARE M, V, N NUMÉRICO
2.  N <- 0
3.  M <- 0
4.  ENQUANTO N <= 10 FAÇA
5.  INÍCIO
6.      ESCREVA “DIGITE O VALOR”
7.      LEIA V
8.      M <- M + V
9.      N <- N + 1
10. FIM
11. M <- M / 10
12. ESCREVA M
```

```c
#include <stdio.h>

int main() {
    float M, V, N;
    N = 0;
    M = 0;
    while (N <= 5) {
        printf("Digite o valor: ");
        scanf("%f", &V);
        M = M + V;
        N = N + 1;
    }
    M = M / 10;
    printf("%f", M);
}
```

- Em pseudocódigo:

```
1. enquanto condição então
2. início
3.         ...
4. fim
```

Em C:

```
1. while (condição) {
2.         ...
3. }
```

Repetição

- While é uma palavra reservada (assim como if, else, etc)
- Uma condição é uma expressão lógica:
  - Ex: x > 10 ou média < 6
- Idéia:

1. Avalia a condição
2. Se for verdadeira, executa toda as instruções dentro do bloco.
   Ao terminar o bloco, verifique novamente a condição.
3. Se a condição não for verdadeira, pula para depois do bloco.

- Fatorial de **n! = n * (n-1)!**

```
1.  DECLARE N, F NUMÉRICO
2.  N <- 0
3.  F <- 1
4.  ESCREVA DIGITE O NUMERO DO FATORIAL
5.  LEIA N
6.  ENQUANTO N > 1 FAÇA
7.  INÍCIO
8.      F <- F * N
9.      N <- N - 1
10. FIM
11. ESCREVA RESULTADO, F
```

```c
#include <stdio.h>

int main() {
    int n = 0;
    int f = 1;

    printf("Digite o número do fatorial: ");
    scanf("%d", &n);

    while (n > 1) {
        f = f * n;
        n = n - 1;
    }

    printf("Resultado: %d", f);
    return 0;
}
```

- Escreva um algoritmo em pseudocódigo e um programa em C que lê um número inteiro $n > 0$ e calcula a soma dos n primeiros termos.

```
1.  DECLARE N, T NUMÉRICO
2.  N <- 0
3.  T <- 0
4.  ESCREVA DIGITE O TAMANHO DA SEQUENCIA
5.  LEIA N
6.  ENQUANTO N > 0 FAÇA
7.  INÍCIO
8.      T <- T + N
9.      N <- N - 1
10. FIM
11. ESCREVA RESULTADO, T
```

```c
#include <stdio.h>

int main() {
    int n = 0;
    int t = 0;

    printf("Digite o tamanho da sequência: ");
    scanf("%d", &n);

    while (n > 0) {
        t = t + n;
        n = n - 1;
    }

    printf("Resultado: %d", t);
    return 0;
}
```

Outra forma de repetição (do while)

- A instrução repita difere da instrução enquanto no sentido que ela inicia a repetição e só ao término do laço ela verifica a condição

```
REPITA
comandos
ATÉ condição
```

Exemplos:

```
X ← 1                  // inicialização da variável X com o valor 1
Y ← 5                  // inicialização da variável Y com o valor 5
REPITA
X ← X + 2              // contador incrementado em 2 unidades
Y ← Y + 1              // contador incrementado em 1 unidade
ATÉ X >= Y
```

REPITA

- É utilizada **quando não se sabe a priori** o número de vezes que um trecho do algoritmo deve ser repetido
  - Também possa ser utilizada quando se conhece esse número.

```
REPITA
comandos
ATÉ condição
```

```
do
 {
   comandos;
 }
while (condição);
```

```
1.  DECLARE N, F NUMÉRICO
2.  N <- 0
3.  F <- 1
4.  ESCREVA DIGITE O NUMERO DO FATORIAL
5.  LEIA N
6.  REPITA
7.  F <- F * N
8.  N <- N - 1
9.  ATÉ N >= 1
10. ESCREVA RESULTADO, F
```

```c
#include <stdio.h>

int main() {
    int n, f;
    n = 0;
    f = 1;
    printf("Digite o Numero do Fatorial: ");
    scanf("%i", &n);
    do {
        f = f * n;
        n = n - 1;
    } while (n >= 1);
    printf("Resultado %i", f);
}
```

Para (for)

- Comando alternativo para o **enquanto e repita**
- Estrutura de condição inicial e repetição são descritas no começo.

INICIALIZAÇÃO CONDIÇÃO ATUALIZAÇÃO

```
soma ← 0
para i ← 1 até n faça passo 1
início
soma ← soma + i
fim
```

Para

- Passo indica como o laço será atualizado
- A omissão indica que será um incremento de um

```
PARA J ← 15 ATÉ 1 FAÇA PASSO -2
ESCREVA J
```

Acumuladores

- Devem ser usados quando a realização de um cálculo precisa de valores obtidos a cada iteração, ou seja, o cálculo só estará pronto com a conclusão da repetição.
- É por isso deve ser inicializado com um valor neutro para a operação em que será utilizado.

Exemplo de acumulador:

```
SOMA ← 0              // inicialização da variável SOMA com o valor zero
PARA I ← 1 ATÉ 5 FAÇA
INÍCIO
 ESCREVA "Digite um número: "
 LEIA NUM
 SOMA ← SOMA + NUM // acumulando o valor da variável NUM na variável SOMA
FIM
ESCREVA "Soma = ",SOMA
```

Para (versão C)

```c
#include <stdio.h>

int main() {
    int soma = 0;
    int i;

    for (i = 1; i <= 1000; i = i + 1) {
        soma = soma + i;
    }

    printf("Soma: %d", soma);
}
```

INICIALIZAÇÃO

CONDIÇÃO

PASSO

For / Para

- Sempre que um for é encontrado:

1. A inicialização é executada;
2. A condição é avaliada: se o resultado da avaliação é verdadeiro, então o bloco de instruções é executado; Se o resultado da avaliação for falso, o programa sai do for.
3. Ao final da execução do bloco de instruções, o passo é executado e o processo todo se repete (voltamos ao passo 2).

Operadores unários

- Para facilitar o uso das estruturas de repetição, a linguagem C apresentam operadores unários de incremento e de decremento
- Operador unário de incremento (**++**)
  - Adiciona uma unidade ao seu operando.
  - Exemplo: **numero++;**
  - É o mesmo que fazer: **numero = numero + 1;**
- Operador unário de decremento (**--**)
  - Subtrai uma unidade de seu operando.
  - Exemplo: **numero--;**
  - É o mesmo que fazer: **numero = numero - 1;**

Somar de 0 até 1000

WHILE

```c
#include <stdio.h>

int main() {
    int soma = 0;
    int i;

    while (i <= 1000) {
        soma = soma + i;
        i++;
    }

    printf("Soma: %d", soma);
}
```

DO WHILE

```c
#include <stdio.h>

int main() {
    int soma = 0;
    int i;
    do {
        soma = soma + i;
        i++;
    } while (i <= 1000);

    printf("Soma: %d", soma);
}
```

FOR

```c
#include <stdio.h>

int main() {
    int soma = 0;
    int i;

    for (i = 1; i <= 1000; i++) {
        soma = soma + i;
    }

    printf("Soma: %d", soma);
}
```

Sair do laço

- É possível sair do laço mesmo que a condição seja válida? Ou no meio dele? Sim!!!
- Para isso utilizamos a instrução SAIR (pseudocódigo) ou BREAK (C)

Sair do Laço

```c
#include <stdio.h>

int main() {
    int i = 0;
    while (i < 5) {
        printf("%i", i);
        if (i == 2) {
            break;
        }
    }
}
```

Continuar / pular rodada do laço

- É possível pular uma rodada do laço usando o comando **continue**
- **Esse comando pula o restante do laço e vai para o seu cabeçalho**

No código ao lado, quando o valor de i é par (i % 2 == 0), o programa executa o comando continue, que faz com que o programa retorne ao cabeçalho.

Note que, o continue irá executar a instrução de passo (i++), caso isso acontecesse, o programa entraria em loop infinito, pois o valor de i não iria aumentar nunca.

```c
#include <stdio.h>

int main() {
    int i = 0;
    for (i = 0; i < 10; i++) {
        if (i % 2 == 0) {
            continue;
        }
        printf("%i\n", i);
    }
}
```

Exercícios

Questão 1. [TRE-RJ] Considere a declaração abaixo, feita em C:

```
for (<<expressão1>>; <<expressão2>>; <<expressão3>>) {
 <<comandos>>;
}
```

Pode-se afirmar que o comando *while* equivalente ao *for* é:

(a) while (<<expressão2>>) { <<expressão1>>; <<comandos>>; <<expressão3>>; };

(b) <<expressão3>>; while (<<expressão2>>) { <<expressão1>>; <<comandos>>; };

(c) <<expressão1>>; while (<<expressão2>>) { <<comandos>>; <<expressão3>>; };

(d) <<expressão1>>; while (<<expressão2>>) { <<comandos>>; } <<expressão3>>;

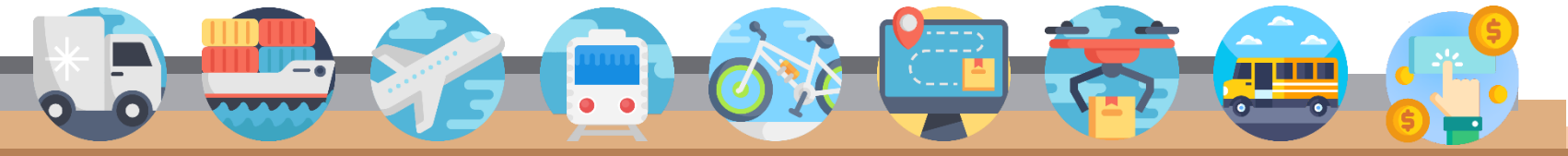
(e) <<expressão3>>; while (<<expressão2>>) { <<comandos>>; <<expressão1>>; };

Exercícios

[CESPE-ABIN] Com relação à linguagens de programação e compiladores, julgue os itens subsequentes como Certo ou Errado. Caso o item esteja errado, justifique e aponte o erro.

(a) “As estruturas de controle de fluxo WHILE e DO...WHILE possuem a mesma finalidade e seus respectivos blocos de comandos são executados pelo menos uma vez em cada uma delas.”

(b) “Nos laços de repetição DO...WHILE e FOR, a condição é verificada no princípio do laço, antes da entrada nesse laço.”



[ConselhoRegionalQuímica] Assinale a alternativa que apresenta as palavras que preenchem, respectivamente, as lacunas do seguinte texto, sobre estruturas de repetição.

O WHILE é uma estrutura de repetição _____, ele repete a execução de um bloco de sentenças enquanto uma condição permanecer verdadeira. Na primeira vez que a condição se tornar falsa, o WHILE _____ a execução do bloco, e a execução continuará com a sentença ou comando que vem logo após _____ WHILE, na sequência do programa. A estrutura de repetição DO ... WHILE tem um comportamento muito semelhante ao WHILE, com uma diferença crucial, a condição é verificada _____ executar o bloco de instruções correspondente.

(a) simples; não repetirá; o bloco do; após.

(b) complexa; repetirá; o; antes.

(c) complexa; não repetirá; o; antes.

(d) simples; repetirá; o bloco do; antes.

(e) complexa; não repetirá; o; após.

[*Adaptado de* PGE-RO] Analise o trecho de código a seguir.

```
1.  int s = 0;
2.  int c = 1;
3.  int d;
4.  while (c < 7) {
5.      d = 4;
6.      while (d > 0) {
7.          s = s + c + d;
8.          d = d - 1;
9.      }
10.     c = c + 1;
11. }
```

Escreva o trecho de código C equivalente utilizando a estrutura de repetição *for*.

1. [Adaptado de IDECAN-IF/CE] Na linguagem C, ao detectar-se entre os comandos inseridos em um *loop*, faz com que ocorra o término imediato da execução do *loop*. Esse comando é conhecido por:

(a) exit

(b) quit

(c) break

(d) halt

Considere o código fonte em linguagem C, abaixo, que utiliza controle de fluxo por meio de um laço de repetição *while* e do comando de tomada de decisão *if*, com objetivo de imprimir uma sequência de números.

```
1.  #include <stdio.h>
2.  int main () {
3.      int cont = 0;
4.      while (cont < 10) {
5.          if (cont == 5) {
6.              continue;
7.          }
8.          printf("%d", cont);
9.          cont = cont + 1;
10.     }
11. }
```

É correto afirmar que o programa está:

(a) correto e será exibida na tela a sequência 0 1 2 3 4 6 7 8 9.

(b) correto e será exibida na tela a sequência 0 1 2 3 4 5 6 7 8 9.

(c) incorreto, pois no lugar da instrução cont = cont + 1; deveria estar cont++

(d) incorreto, pois o bloco if (cont == 5) continue; fará com que o laço se torne infinito.

[FGV-2014] Analise, a seguir, um pequeno programa desenvolvido em C.

```
1.  #include <stdio.h>
2.  int main() {
3.      int i, j, saida;
4.      i = 1;
5.      saida = 0;
6.      while (i < 10) {
7.          if (i % 2 == 0) {
8.              j = 0;
9.          } else {
10.             j = i;
11.         }
12.         saida = saida + j;
13.         i = i + 1;
14.     }
15.     printf("%d", saida);
16. }
```

O valor da variável saída que é impresso na tela ao final da execução do programa é

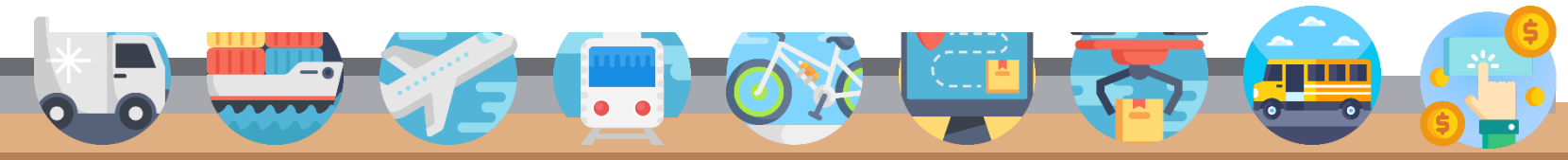
(a) 30

(b) 26

(c) 25

(d) 22

(e) 20



Desafio

- A Subindo Bem Confortavelmente (SBC) é uma empresa tradicional, com mais de 50 anos de experiência na fabricação de elevadores. Todos os projetos da SBC seguem as mais estritas normas de segurança, mas infelizmente uma série de acidentes com seus elevadores manchou a reputação da empresa.
- Ao estudar os acidentes, os engenheiros da companhia concluíram que, em vários casos, o acidente foi causado pelo excesso de passageiros no elevador. Por isso, a SBC decidiu fiscalizar com mais rigor o uso de seus elevadores: foi instalado um sensor em cada porta que detecta a quantidade de pessoas que saem e entram em cada andar do elevador. A SBC tem os registros do sensor de todo um dia de funcionamento do elevador (que sempre começa vazio). Eles sabem que as pessoas são educadas e sempre deixam todos os passageiros que irão sair em um andar saírem antes de outros passageiros entrarem no elevador, mas ainda assim eles têm tido dificuldade em decidir se a capacidade máxima do elevador foi excedida ou não.
- Sua tarefa é escrever um programa que, dada uma sequência de leituras do sensor e a capacidade máxima do elevador, determinar se a capacidade máxima do elevador foi excedida em algum momento.

Entrada

- A primeira linha da entrada contém dois inteiros **N** e **C**, indicando o número de leituras realizadas pelo sensor e a capacidade máxima do elevador, respectivamente ($1 \leq \mathbf{N} \leq 1000$ e $1 \leq \mathbf{C} \leq 1000$). As **N** linhas seguintes contém, cada uma, uma leitura do sensor. Cada uma dessas linhas contém dois inteiros **S** e **E**, indicando quantas pessoas saíram e quantas pessoas entraram naquele andar, respectivamente ($0 \leq \mathbf{S} \leq 1000$ e $0 \leq \mathbf{E} \leq 1000$).

Saída

- Seu programa deve imprimir uma uníca linha contendo o caractere ‘S’, caso a capacidade do elevador tenha sido excedida em algum momento, ou o caractere ‘N’ caso contrário.

Exemplos de Entrada	Exemplos de Saída
5 10 0 5 2 7 3 3 5 2 7 0	N
5 10 0 3 0 5 0 2 3 4 6 4	S

Desafio

- https://www.beecrowd.com.br/judge/pt/problems/view/1847

Obrigado!

Perguntas?

Marcos Roriz (marcosroriz@ufg.br)

ENGENHARIA DE TRANSPORTES

FCT
FACULDADE DE CIÊNCIAS E TECNOLOGIA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Fonte dos ícones utilizados: flaticons.com

Obrigado!

Perguntas?

Marcos Roriz (marcosroriz@ufg.br)

ENGENHARIA DE TRANSPORTES

FCT
FACULDADE DE CIÊNCIAS E TECNOLOGIA

UFG
UNIVERSIDADE FEDERAL DE GOIÁS

Fonte dos ícones utilizados: flaticons.com