Introdução à Programação

1.Ano LCC-MIERSI
DCC - FCUP

Nelma Moreira

Aula 2

Etapas para o desenvolvimento dum programa

1. Perceber o problema
2. Encontrar um procedimento algorítmico para o resolver. Estratégias:
    1. Determinar uma sequência de passos elementares.
    2. Dividir o problema em sub-problemas e para cada um deles usar a estratégia anterior.
3. Formular o algoritmo e representá-lo por um programa
4. Avaliar a correcção do programa

Construir um tutor que ajude a aprendizagem da aritmética para alunos do primeiro ciclo.

Deve-se ser mais concreto: por exemplo, tutor para a adição de dois inteiros.

Pensar como é que nós resolvemos o problema:

1. Ler dois inteiros
2. Calcular a sua soma
3. Ler a resposta do aluno.
4. Comparar os dois valores. Se o valor do aluno estiver correcto, escrever uma mensagem de congratulações, senão informar que o resultado não está correcto.

O algoritmo pode-se escrever em pseudo-código como:

```
Ler dois inteiros e guardá-los em n1 e n2

Adicionar os valores de n1 e n2 e guardar o resultado
em sol

Ler a resposta e guardá-la em res

Se o valor de sol é igual ao de res então  escrever
"Parabéns, senão  escrever "Enganaste-te! Tenta outra
vez!"
```

ou simplificando a notação

```
Ler n1, n2
sol = n1 + n2
Ler res
Se sol == res então  escrever "Parabéns"
                senão  escrever "Enganaste-te! Tenta
outra vez!"
```

```c
#include <stdio.h>
main() {
  int n1, n2, sol, res;
  printf("Introduz o primeiro inteiro\n");
  scanf("%d", &n1);
  printf("Introduz o segundo  inteiro\n");
  scanf("%d",&n2);
  sol=n1+n2;
  printf("Introduz a tua resposta\n");
  scanf("%d",&res);
  if (sol==res) printf("Parabens!!!\n");
  else printf("Enganaste-te! Tenta outra vez!\n");
}
```

Modificações

* Aprender a multiplicar
* Se a resposta não estiver correcta ser dada a resposta certa.
* Permitir o aluno introduzir vários pares de números

Permitir o aluno introduzir vários pares de números

```
conta = 5 (ou Ler conta)

Enquanto conta > 0 faça

    Ler n1, n2
    sol = n1 + n2
    Ler res
    Se sol == res então  escrever "Parabéns"
                 senão  escrever "Errado!"
    conta = conta - 1
```

Pseudo-código

* Leitura: passagem de informação do exterior para o programa. Ex: `Ler n1, n2`

  Os valores lidos ficam guardados, respectivamente em `n1` e `n2`.

* Escrita: passagem de informação para o exterior. Ex: `Escrever n2`

  O valor guardado em `n2` é escrito (no ecrã ou num ficheiro

Pseudo-código

* Atribuição: Determinar um valor e guardá-lo.

  Ex: `sol=n1+n2`

* Condicional

Se `condição` **então** *`acção1`* **senão** `acção2`

Se a `condição` **for verdadeira executa** `acção1` **, se for falsa executa** `acção2`.

Pseudo-código

* Ciclo

Enquanto `condição` **faça** `acção`

Um ou vários passos são repetidos enquanto uma condição é verificada.

Problema 2

Escreve um programa em C que dados dois inteiros indique se são iguais ou qual o maior.

5 Etapas

1. Perceber o problema
2. Ideia da resolução
3. Algoritmo
4. Codificação em C
5. Verificação

Pseudo-código P2

1. Ler um inteiro e guardar em n1
2. Ler um inteiro e guardar em n2
3. Se n1 = = n2, escrever n1 é igual a n2
4. Se n1 < n2, escrever n2 é maior do que n1
5. Se n1 > n2, escrever n1 é maior do que n2

Programa #2 em C

```
#include <stdio.h>
main() {
   int n1,n2;
   printf("Introduz o primeiro inteiro\n");
   scanf("%d", &n1);
   printf("Introduz o segundo  inteiro\n");
   scanf("%d",&n2);
   if (n1 == n2)
     printf("%d igual a %d\n", n1,n2);
   else if (n1 < n2)
     printf("%d maior do que a %d\n", n2,n1);
   else
     printf("%d maior do que a %d\n", n1,n2);
}
```

Notas sobre C

* O programa tem pelo menos uma função: `main()`
  Esta função tem sempre que existir!

* `n1` e `n2` são variáveis que estão declaradas com um tipo:
  `int` inteiro
  As variáveis são "caixas" que permitem guardar valores.

* Cada instrução termina com um `;` (ponto-e-vírgula)

* A mudança de linha não tem significado especial

* A identação das linhas não tem significado mas é normalmente usada para tornar o programa mais legível

Notas sobre C

* Uma instrução `if` permite a execução condicional de instruções

```
if (condicao) instrucao1 [else instrucao2]
```

* `instrucao1` é executada se a `condicao` for verdadeira. Se for falsa, `instrucao2` é executada se existir.

Operadores relacionais: `==, !=, <, <=, >, >=...`

Notas sobre C

* As chamadas à função `printf()`, escrevem a mensagem entre aspas representando `\n` a mudança de linha.
* A chamada à função `scanf("%d",&n2)`, permite a introdução (leitura) de um valor, como inteiro decimal, que é guardado na variável `n2`.
* Ambas as funções pertencem à biblioteca standard do C.: `# include <stdio.h>`

Executar um programa em C

* Escrever o programa usando um editor de texto (por exemplo, `Emacs`) e guardar num ficheiro com a terminação `.c` Ex: `comp.c`

* Compilar o programa com o compilador `gcc` ou `cc`

```
$ gcc comp.c -o comp
```

O programa executável chama-se `comp`
(Sem a opção `-o` o executável é `a.out`)

Executar um programa em C

* Executar o programa (comando): numa `shell` basta escrever o nome dele:

```
$ comp
Introduz o primeiro inteiro
23
Introduz o segundo  inteiro
45
45 é maior do que 23
$
```

Problema 3

A qualidade do ar é medida por um índice numérico. Se o índice for menor que 35 a qualidade do ar é Boa. Se esse valor for entre 35 e 60 é Má. Se esse valor for maior do que 60, a qualidade é Péssima.

Etapas

1. Ok...
2. Comecemos por analizar um só valor:
3. Ler um índice
4. Escrever a mensagem apropriada, baseada no valor do índice.

Algoritmo

```
Ler indice

Se indice < 35 então escrever "Boa"
   senão se indice >=35 e indice <=
60 escrever "Má"
   senão escrever "Péssima"
```

Programa #3 em C

```
#include <stdio.h>
main() {
   int indice;
   printf("Indice da qualidade do ar:");
   scanf("%d",&indice);
   if (indice < 35)   printf("Boa\n");
   else if( indice >= 35 && indice <= 60)  printf("Má\n");
   else printf("Péssima\n");
}
```

Problema 4 **

Analisar a qualidade do ar ao longo de 30 dias. Deve-se determinar o número de dias com cada classe de qualidade do ar e qual a média aritmética ao fim do mês.

Etapas

1. ...

2. Considerar alguns valores: 30, 45, 40, 38, 49, 55, 40, 34,... e resolver o problema à mão. Descrever o que foi feito: Para cada dia, ler um índice, determinar a sua classe e incrementar o número de dias com essa classe. Ir somando os valores lidos para determinar a média no fim.

Etapas

Variáveis necessárias (caixas)

`indice` para o índice

`dias` para contar os dias `1` a `30`...

`boa` para contar quantos os dias com qualidade do ar "Boa"; valor inicial `0`.

`ma` para contar quantos os dias com qualidade do ar "Má", valor inicial `0`.

`pessima` para contar quantos os dias com qualidade do ar "Péssima", começa em `0`.

`soma` para o cálculo da média é necessário somar todos os valores dos índices; começa em `0`.

`media` para a média

Algoritmo

```
dias = 1, boa = 0, ma = 0, pessima = 0, soma = 0
Enquanto dias <= 30 faça
  Ler indice
  Se indice < 35 então boa = boa + 1
     senão se indice >=35 e indice <= 60 ma = ma + 1
              senão pessima = pessima + 1
     soma = soma + indice
     dias = dias + 1
media = soma / 30
Escrever boa, ma, pessima, media
```

Programa #4 em C

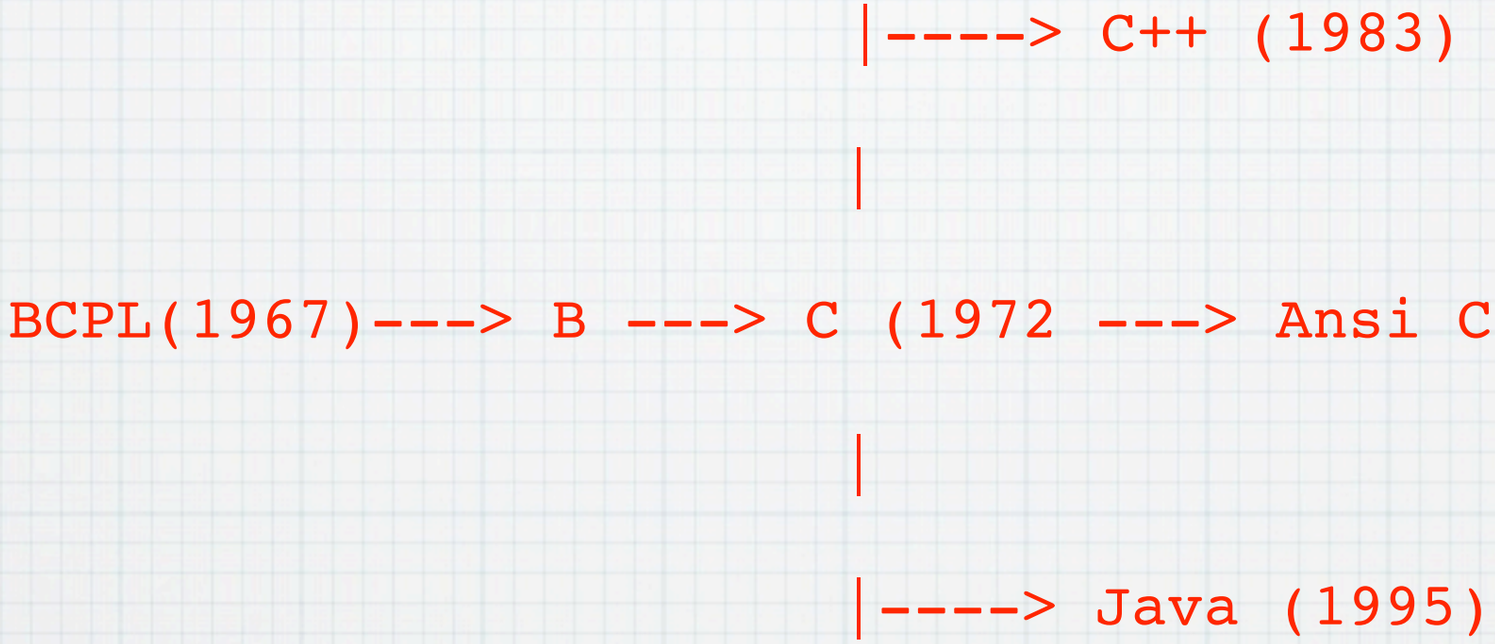
```
#include <stdio.h>
#define DIAS 30
main() {
   int indice, dias = 1, boa = 0, ma = 0, pessima = 0,
soma = 0;
   double media;
   printf("Indices da qualidade do ar:\n");
   while (dias <= DIAS) {
        scanf("%d",&indice);
        if (indice < 35) boa = boa + 1;
        else if( indice >= 35 && indice <= 60)  ma = ma +
1;
        else pessima = pessima + 1;
        soma = soma + indice;
        dias = dias + 1;
    }
    media = (float) soma / DIAS;

  ....FALTAM OS PRINTFS!!!
}
```

...os PRINTFS..

```
printf("\nQualidade || Número de dias\n");
printf("-----------------------------\n\n");
printf("   Boa              %4d\n",boa);
printf("   Ma               %4d\n",ma);
printf(" Pessima            %4d\n\n",pessima);
printf("Em média a qualidade foi ");
if (media < 35 ) printf("Boa\n");
else if( media >= 35 && media <= 60)  printf("Má\n");
      else printf("Péssima\n");
}
```

C



O quê?

* C é linguagem com estruturas de alto nível mais perto da linguagem máquina em termos de eficiência...

* É usada essencialmente para tarefas em que a velocidade é crítica: interfaces a sistemas de operação; todo o tipo de algoritmos com "complexidade" alta.

Estrutura de um Programa em C

* Comandos do preprocessador para incluir ficheiros (`#include`)
* Definição de constantes (`#define`)
* Declaração de variáveis
* Definição de funções
* Definição da função `main()`

Tipos, Expressões,Operadores

* Tipos de dados
* Variáveis
* Constantes
* Constantes Simbólicas
* Expressões
* Operadores aritméticos, relacionais e lógicos
* Operadores de atribuição, incrementais, atribuição artméticos
* Expressões condicionais

Instruções (imperativas)

* Atribuições
* Instrução composta (sequêncial)
* Instruções de Controlo:
  * Condicionais
  * Ciclos