Capítulo 2. Variáveis do tipo INTEIRO

17 Mar 2025

OBJETIVOS DO CAPÍTULO

- Conceitos de: variáveis do tipo inteiro, atribuição, avisos e erros de compilação, erros de execução, comentários dentro do programa-fonte
- Operadores matemáticos básicos
- Novos comandos do FORTRAN a usar: INTEGER e READ

Para inicializar as atividades deste capítulo, deve-se executar o programa Fortran.

2.1 programa02.f90, versão A

1) No Fortran, seguindo o procedimento do Capítulo 1, **criar um projeto** com o nome **programa02**
2) No Fortran, seguindo o procedimento do Capítulo 1, **criar e inserir** no projeto o programa-fonte **programa02.f90**
3) Conforme é mostrado na Figura 2.1, dentro do espaço de edição do Fortran, na subjanela maior, copiar exatamente o texto mostrado na Tabela 2.1, abaixo, que está em vermelho.

Tabela 2.1 programa02.f90, versão A

```
INTEGER A
WRITE(*,*) "A"
WRITE(*,*) A
END
```

4) Comentários sobre o programa:
   a) No capítulo 1 foram usados os comandos WRITE e END da linguagem FORTRAN. No programa02.f90, há um novo comando: INTEGER. Ele é usado para definir variáveis do tipo inteiro, isto é, variáveis que podem guardar ou armazenar na memória do computador números inteiros, positivos ou negativos, como 2, 5, 0, 54367 ou –3. Os valores inteiros que podem ser usados vão de -2^{31} a $2^{31}-1$, que é 2,147,483,647.
   b) A linha `INTEGER A` define a variável A como sendo do tipo inteiro. Este comando reserva um espaço na memória do computador, utilizando o nome ou rótulo A para armazenar um valor inteiro.
   c) A linha `WRITE(*,*) "A"` escreve o comentário que está entre aspas; no caso a letra A.
   d) A linha `WRITE(*,*) A` escreve o valor da variável A que está armazenado na memória do computador.
   e) A linha `END` encerra o programa.
5) Ao se compilar o programa, executando **Build, Compile**, o resultado deve ser o mostrado na Figura 2.1. Deve-se notar na subjanela inferior um aviso (warning) mencionando que o valor da variável A não foi

definido. Geralmente, é possível ver mais informações sobre cada código de erro ou aviso do Fortran executando **? InfoView, Build Errors, localizar o erro específico**; ver, por exemplo, o FOR4265.

6) Gerar o programa-executável fazendo **Build, Build**.

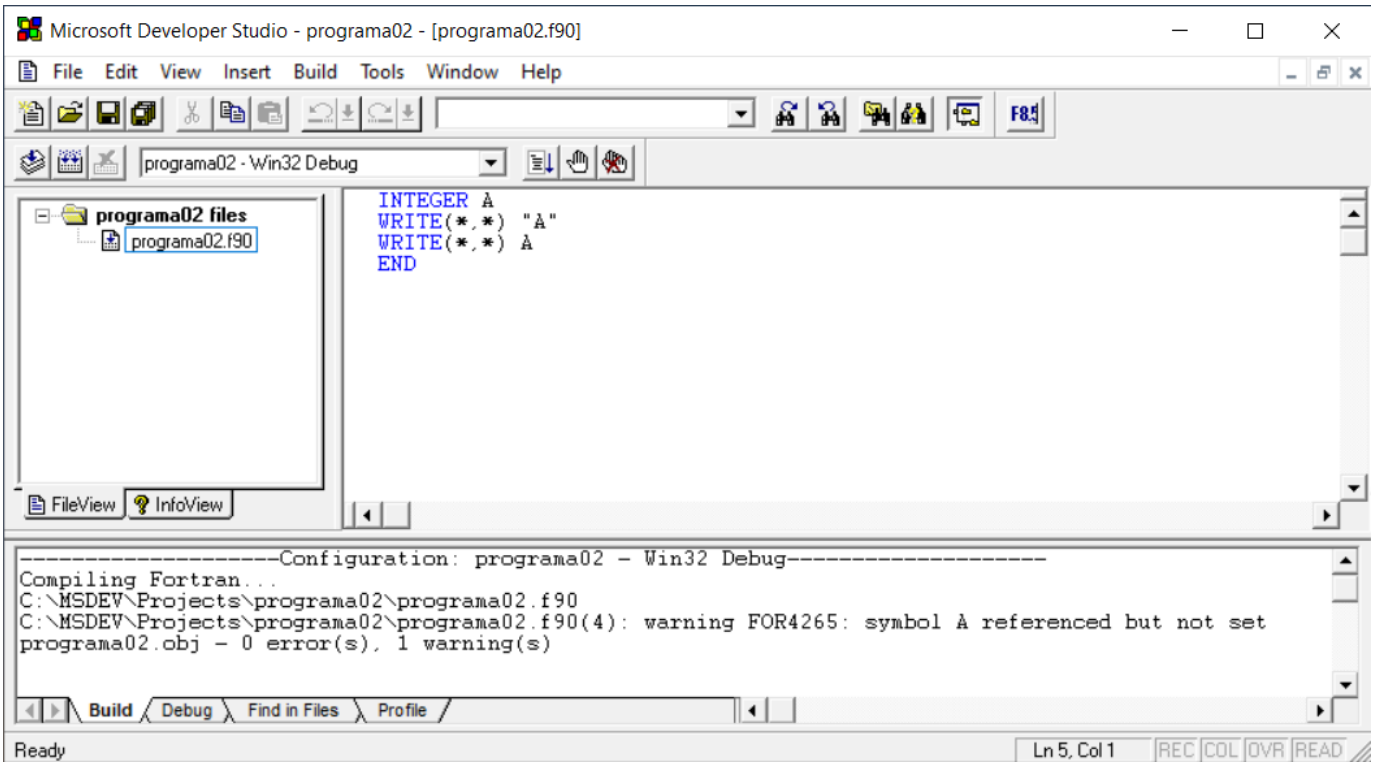

Figura 2.1 Resultado da compilação do programa02.f90, versão A.

7) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 2.2, dentro da qual tem-se:

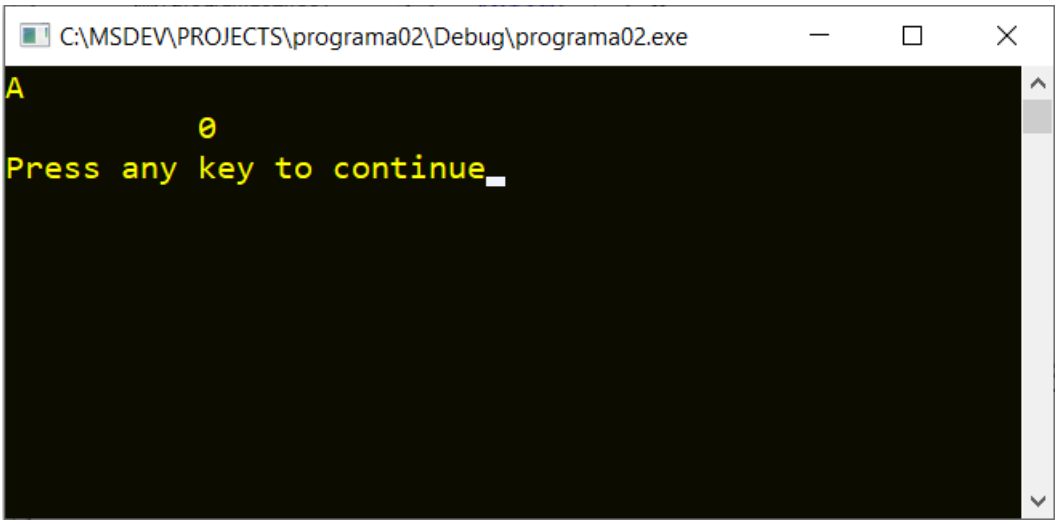

Figura 2.2 Resultado da execução do programa02.f90, versão A.

a) Na primeira linha, a letra A, resultado do comando `WRITE(*,*) "A"` do programa.

b) Na segunda linha, o valor zero, resultado do comando `WRITE(*,*) A` do programa. Isso ocorre porque não foi definido um valor para a variável A, conforme o próprio Fortran informou durante a compilação do programa. Portanto, sempre é necessário definir o valor de cada variável do programa, caso contrário, por default deste compilador FORTRAN, assume-se valor nulo.

c) E na terceira linha, a frase "Press any key to continue". Como diz este aviso, basta clicar em qualquer tecla para continuar. Ao se fazer isso, a execução do programa é encerrada.

8) Deve-se perceber a diferença que existe entre os comandos `WRITE(*,*) "A"` e `WRITE(*,*) A` do programa. No primeiro, escreve-se a letra A que é um comentário. No segundo, escreve-se o valor da variável A que está guardado na memória do computador.

2.2 programa02.f90, versão B

1) Dentro do Fortran, usando o mesmo programa-fonte da versão A, editar exatamente o texto mostrado na Tabela 2.2, abaixo, que está em vermelho.

Tabela 2.2 programa02.f90, versão B

```
INTEGER A
A = 3
WRITE(*,*) "A"
WRITE(*,*) A
END
```

2) Comentários sobre o programa: a única diferença entre a versão anterior (A) e a atual (B) do programa02.f90 é a inclusão da segunda linha, ou seja, `A = 3`. O sinal de igualdade dentro de um programa escrito em linguagem FORTRAN é utilizado para atribuir o valor que está do lado direito à variável do lado esquerdo. Portanto, neste caso, o valor 3 é atribuído à variável A. Em outras palavras, o valor 3 é armazenado no espaço da memória do computador que é identificado pelo nome ou rótulo A, o nome da variável. Este valor utilizado (3) é apenas um exemplo; ele pode ser qualquer número inteiro permitido.

3) Nesta versão do programa, ao se executar **Build, Compile** não haverá aviso (warning) porque, neste caso, o valor da variável A está definido.

4) Gerar o programa-executável fazendo **Build, Build**.

5) Ao se executar o programa através de **Build, Execute** surge uma janela, mostrada na Figura 2.3, dentro da qual tem-se:

a) Na primeira linha, a letra A, resultado do comando `WRITE(*,*) "A"` do programa.

b) Na segunda linha, o valor 3, resultado do comando `WRITE(*,*) A` do programa e do comando `A = 3`.

c) E na terceira linha, a frase "Press any key to continue".

6) Deve-se perceber que o programa é executado, linha por linha, da primeira (`INTEGER A`) até a última (`END`), de cima para baixo.

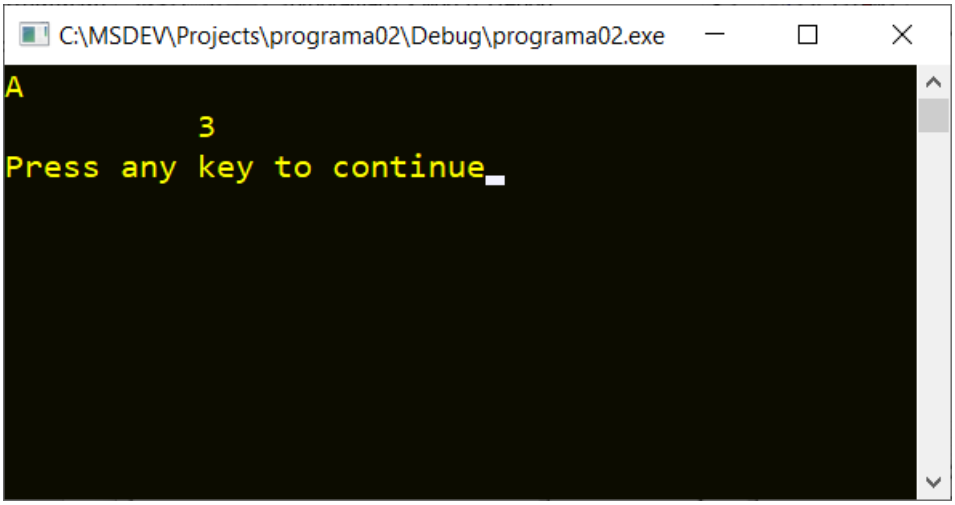

Figura 2.3 Resultado da execução do programa02.f90, versão B.

7) Um exemplo proposital de erro de compilação é apresentado na Figura 2.4. Neste caso, ele ocorre devido à eliminação do segundo asterisco da terceira linha do programa. Erros de compilação ocorrem quando os comandos do FORTRAN são utilizados com sintaxe incorreta. Na subjanela inferior do Fortran, geralmente haverá um comentário indicando o código de cada erro (error). Logo após o nome do programa-fonte compilado, entre parênteses, é indicado o número da linha do programa-fonte onde deve estar o erro. No exemplo da Figura 2.4, o compilador do Fortran mostra o seguinte comentário:
C:\MSDEV\Projects\programa02\programa02.f90(3): error FOR3852: syntax error detected between , and)
Portanto, este comentário indica que na linha 3 há um erro de sintaxe (erro que resulta do uso incorreto de um comando, no caso o WRITE) entre a vírgula e o símbolo de fecha parênteses. Em geral, fazendo-se um duplo clique sobre a mensagem de erro, uma seta indica a linha em que o erro se encontra.

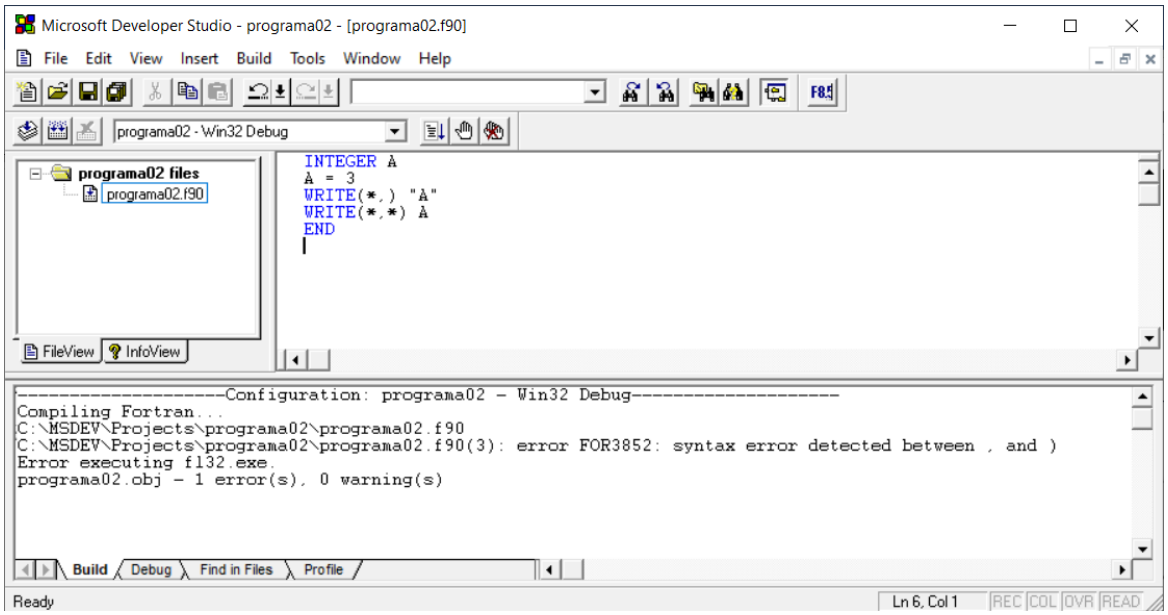

Figura 2.4 Exemplo de erro de compilação.

8) As linhas do programa-fonte são numeradas do topo para baixo, e as colunas, da esquerda para a direita. Na extremidade inferior da janela do Fortran, do lado direito, sempre são indicadas a linha (Ln) e a coluna (Col) onde o cursor se encontra dentro do programa-fonte.

2.3 programa02.f90, versão C

1) Dentro do Fortran, usando o programa-fonte da versão B, editar exatamente o texto mostrado abaixo na Tabela 2.3, que está em vermelho.

Tabela 2.3 programa02.f90, versão C

```
INTEGER A
A = 3
WRITE(*,*) "Valor de A = ", A
END
```

2) Comentários sobre o programa: a diferença entre a versão anterior (B) e a atual (C) do programa02.f90 é a junção dos dois comandos WRITE em apenas um, na terceira linha do programa, isto é, `WRITE(*,*) "Valor de A = ", A`. Esta forma do comando WRITE é usada quando se quer escrever na mesma linha dois ou mais elementos. No caso, são apenas dois elementos, ou seja, o comentário `Valor de A =` e a variável `A`. Os elementos devem ser separados por vírgula.
3) Executar **Build, Compile** para compilar o programa.
4) Gerar o programa-executável fazendo **Build, Build**.
5) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 2.5, dentro da qual tem-se:

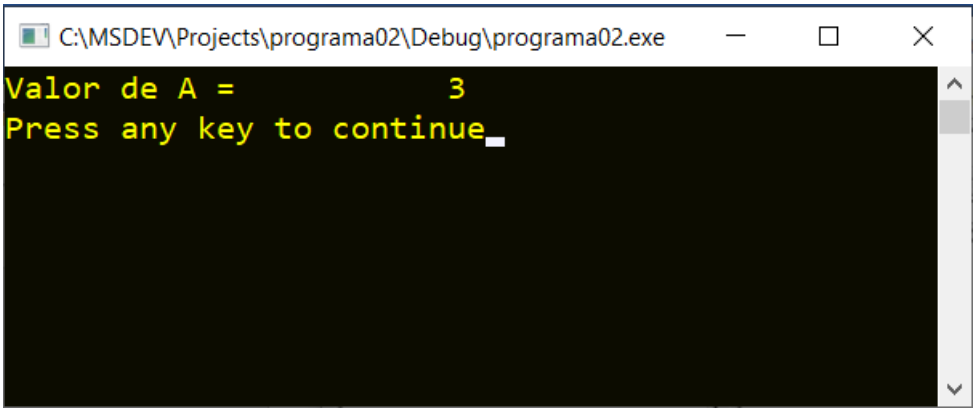

Figura 2.5 Resultado da execução do programa02.f90, versão C.

a) Na primeira linha, como resultado do comando `WRITE(*,*) "Valor de A = ", A` do programa, o comentário Valor de A = e, na mesma linha, o valor da variável A, cujo valor atribuído dentro do programa

foi 3. Esta forma é mais adequada para identificar uma variável e o seu valor. Os espaços entre o sinal de igualdade e o valor 3 ocorrem por default. A manipulação destes espaços será vista no Capítulo 6.

b) E na segunda linha, a frase Press any key to continue.

2.4 programa02.f90, versão D

1) Dentro do Fortran, usando o programa-fonte da versão C, editar exatamente o texto mostrado abaixo na Tabela 2.4, que está em vermelho.

Tabela 2.4 programa02.f90, versão D

```
INTEGER A
A = 4
A = 3
WRITE(*,*) "Valor de A = ", A
END
```

2) Comentários sobre o programa: a única diferença entre a versão anterior (C) e a atual (D) do programa02.f90 é a inclusão da linha `A = 4`, que atribui o valor 4 à variável A.
3) Executar **Build, Compile** para compilar o programa.
4) Gerar o programa-executável fazendo **Build, Build**.
5) Ao se executar o programa, através de **Build, Execute**, o resultado é o mesmo já mostrado na Figura 2.5. Isso ocorre porque o programa é executado, linha por linha, da primeira (`INTEGER A`) até a última (`END`), de cima para baixo. Assim, embora tenha sido atribuído o valor 4 à variável A na segunda linha do programa, na linha seguinte atribui-se o valor 3 à mesma variável, e, na quarta linha do programa, escreve-se o valor de A. O valor escrito é 3 porque é o último valor que foi armazenado na memória do computador. A denominação "variável" é usada justamente porque seu valor pode ser alterado ao longo da execução do programa.

2.5 programa02.f90, versão E

1) Dentro do Fortran, usando o programa-fonte da versão D, editar exatamente o texto mostrado abaixo na Tabela 2.5, que está em vermelho, incluindo a linha em branco.

Tabela 2.5 programa02.f90, versão E

```
! Programa02.f90
INTEGER A ! velocidade

A = 3 ! atribui o valor 3 à variável A
WRITE(*,*) "Valor de A = ", A, " m/s"
END
```

2) Comentários sobre o programa: em cada linha do programa-fonte, tudo que estiver à direita do símbolo ! (exclamação) não é executado pelo programa. São apenas comentários usados para esclarecer o que faz cada parte do programa. Isso é chamado de documentação interna. Dentro do editor do Fortran, todos os comentários ficam na cor verde, como é mostrado na Figura 2.6. Um comentário pode envolver uma linha inteira do programa, como na primeira da versão E, ou apenas uma parte, como na quarta linha do programa. Linhas em branco dentro do programa-fonte também não são executadas, elas equivalem a um comentário em branco. Um exemplo é a terceira linha do programa02.f90, versão E, na Figura 2.6. No comando WRITE há 3 elementos.
3) Executar **Build, Compile** para compilar o programa.
4) Gerar o programa-executável fazendo **Build, Build**.
5) Ao se executar o programa, através de **Build, Execute**, o resultado é o mesmo mostrado na Figura 2.5 acrescido de "m/s" após o valor 3. Isso ocorre porque as diferenças entre a versão C e a atual (E) do programa02.f90 são apenas os comentários e uma linha em branco, que não são executados pelo programa, e o acréscimo do terceiro elemento no WRITE.
6) Para maior clareza e facilidade de compreensão do programa-fonte, recomenda-se que dentro dele sejam usados comentários e linhas em branco.

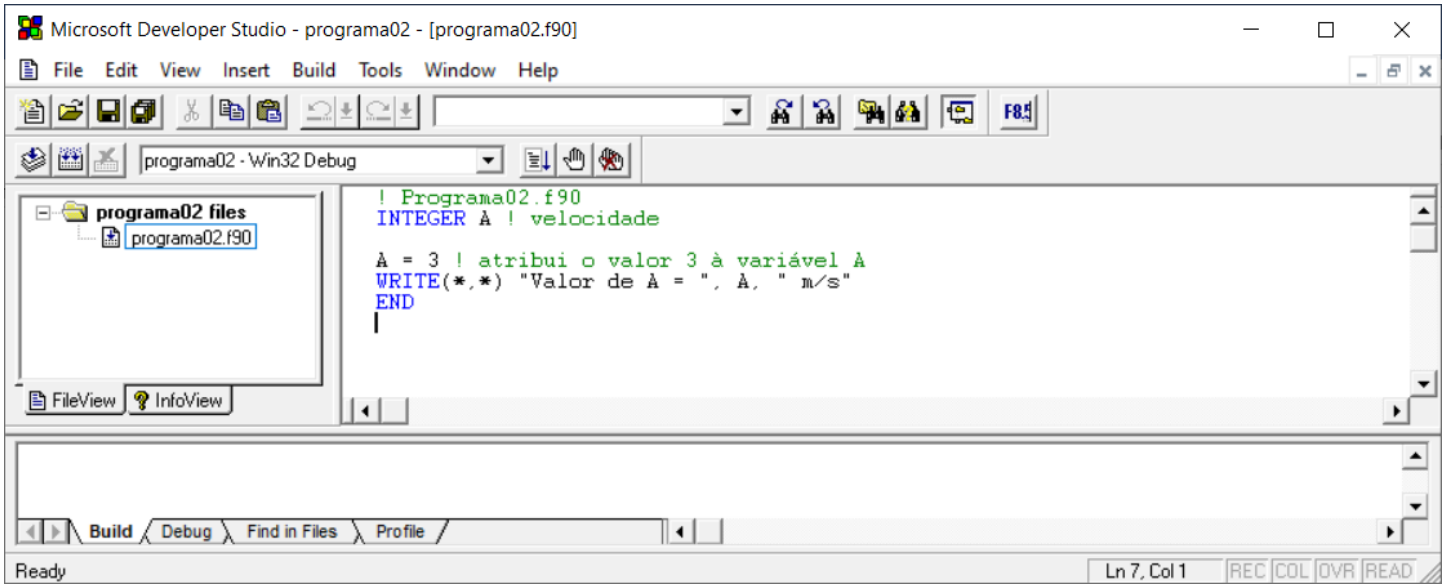

Figura 2.6 Programa02.f90, versão E.

2.6 programa03.f90, versão A

Para inicializar as atividades com o próximo programa, deve-se executar:

1) No Fortran, para fechar o projeto atual: **File, Close Workspace**
2) Na janela Microsoft Developer Studio, clicar sobre o botão **Sim**
3) No Fortran, seguindo o procedimento do Capítulo 1, **criar um projeto** com o nome **programa03**

4) No Fortran, seguindo o procedimento do Capítulo 1, **criar e inserir** no projeto o programa-fonte **programa03.f90**
5) Dentro do Fortran, editar exatamente o texto mostrado em vermelho na Tabela 2.6, abaixo, incluindo as linhas em branco. Comentar sobre o uso da tecla "Insert" do teclado numérico, que fica na mesma tecla do número zero: se ela estiver acionada, consegue-se inserir linhas e caracteres no programa-fonte, por exemplo, caso contrário, não; assim, se ela não estiver acionada, ao digitar novos caracteres, eles serão sobrescritos aos outros já existentes.

Tabela 2.6 programa03.f90

```
INTEGER A, B, C, D, E, F, G

A = -6
B = 2

C = A + B
D = B - A
E = A * B
F = A / B
G = A ** B

WRITE(*,*) "A = ", A
WRITE(*,*) "B = ", B
WRITE(*,*) "C = A + B = ", C
WRITE(*,*) "D = B - A = ", D
WRITE(*,*) "E = A * B = ", E
WRITE(*,*) "F = A / B = ", F
WRITE(*,*) "G = A ** B = ", G

END
```

6) Comentários sobre o programa:
   a) A linha `INTEGER A, B, C, D, E, F, G` define as variáveis A, B, C, D, E, F e G como sendo do tipo inteiro. Este comando reserva espaço na memória do computador para diversas variáveis com apenas um comando INTEGER. Entretanto, as variáveis devem estar separadas por vírgula.
   b) As linhas `A = -6` e `B = 2` atribuem os valores inteiros –6 e 2 às variáveis A e B. Estes valores são apenas exemplos; eles podem ser quaisquer números inteiros.
   c) As variáveis C, D, E, F e G são calculadas em função dos valores das variáveis A e B, usando os cinco operadores matemáticos básicos definidos na Tabela 2.7, conforme explicado a seguir.
   d) A linha `C = A + B` adiciona os valores das variáveis A e B e atribui o resultado à variável C. Esta instrução corresponde à seguinte expressão algébrica: C = A + B

e) A linha `D = B - A` subtrai o valor da variável A do valor da variável B e atribui o resultado à variável D. Esta instrução corresponde à seguinte expressão algébrica: D = B – A

f) A linha `E = A * B` multiplica os valores das variáveis A e B e atribui o resultado à variável E. Esta instrução corresponde à seguinte expressão algébrica: E = A B

g) A linha `F = A / B` divide o valor da variável A pelo valor da variável B e atribui o resultado à variável F. Esta instrução corresponde à seguinte expressão algébrica: $F = \frac{A}{B}$

h) A linha `G = A ** B` eleva o valor da variável A à potência definida pelo valor da variável B e atribui o resultado à variável G. Esta instrução corresponde à seguinte expressão algébrica: $G = A^B$

Tabela 2.7 Operadores matemáticos básicos em FORTRAN.

Operação matemática	Símbolo do operador matemático em FORTRAN	Nome do símbolo
Adição	+	mais
Subtração	–	menos
Multiplicação	*	asterisco
Divisão	/	barra
Potenciação	**	duplo asterisco

7) Executar **Build, Compile** para compilar o programa.

8) Gerar o programa-executável fazendo **Build, Build**.

9) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 2.7, dentro da qual tem-se: os valores das variáveis A e B; e os valores resultantes das cinco operações matemáticas básicas da Tabela 2.7 efetuadas com as variáveis A e B. **Analisar** cada resultado, comparando-o com o valor esperado obtido de um cálculo mental.

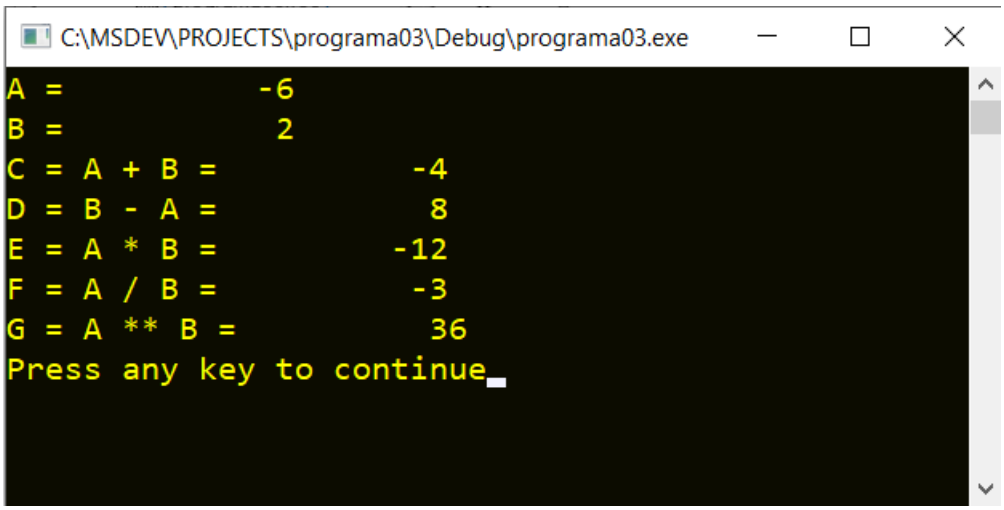

Figura 2.7 Resultado da execução do programa03.f90, versão A, com A = –6 e B = 2.

10) **Dentro do programa-fonte, fazer A = 1 e B = 2**. **Compilar** e **gerar o executável** do programa com esta alteração. **Executar o programa** cujo resultado deve ser aquele mostrado na Figura 2.8. **Analisar** cada resultado, comparando-o com o valor esperado obtido de um cálculo mental. O resultado da divisão pode parecer incorreto, mas não é. Isso se deve ao seguinte: o valor que resulta de um cálculo envolvendo dois números inteiros também é um número inteiro, que corresponde à parte inteira do número real equivalente ao cálculo realizado. Portanto, o resultado de 1 dividido por 2 resulta em 0,5, que é um número real. Mas como o cálculo envolve dois números inteiros, a parte inteira do número real 0,5 é 0, que é o resultado da divisão mostrado na Figura 2.8. Este tema será explorado com mais detalhes no Capítulo 3.

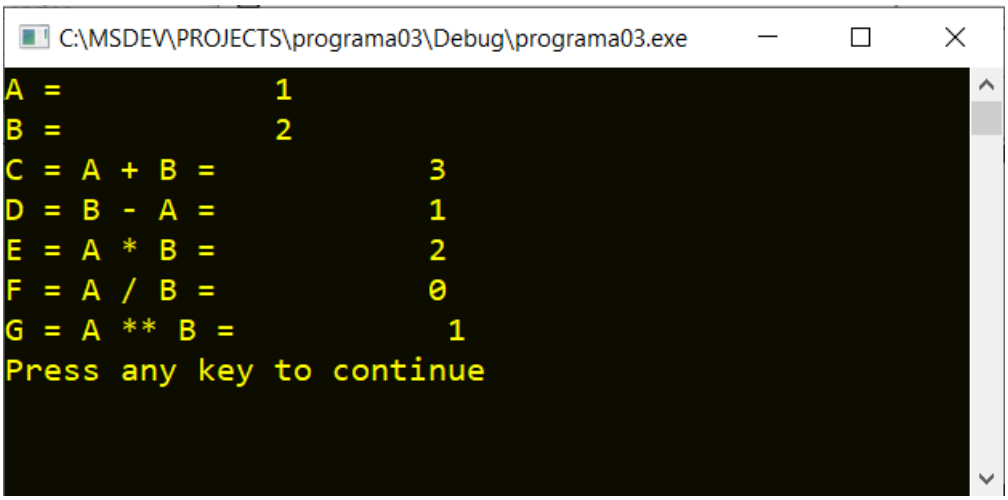

Figura 2.8 Resultado da execução do programa03.f90, versão A, com A = 1 e B = 2.

11) Para confirmar a afirmação anterior, **dentro do programa-fonte, fazer A = 5 e B = 3**. **Compilar** e **gerar o executável** do programa com esta nova alteração. **Executar o programa**. **Analisar** cada resultado, comparando-o com o valor esperado obtido de um cálculo mental; atenção ao resultado da divisão, que é 1.
12) Atribuir valores às variáveis dentro do próprio programa-fonte não é recomendável. Pois, para alterar os valores, é necessário ter o programa-fonte, além de recompilá-lo e gerar o programa-executável a cada vez. O procedimento mais indicado é utilizar o comando READ, apresentado na próxima seção.

2.7 programa03.f90, versão B

1) Dentro do Fortran, partindo da versão A do programa-fonte programa03.f90, editar exatamente o texto mostrado em vermelho na Tabela 2.8, abaixo, incluindo as linhas em branco.
2) Comentários sobre o programa:
   a) A única diferença entre a versão anterior (A) e a atual (B) do programa03.f90 está no ínicio do programa. É a inclusão de 4 linhas novas no lugar de se atribuir valores às variáveis A e B.
   b) Até aqui, os comandos da linguagem FORTRAN que foram usados são: WRITE, END e INTEGER. Na versão B do programa03.f90, há um novo comando: READ. Ele é usado para atribuir (fornecer) valores

às variáveis durante a execução de um programa. Isto é, o comando READ é empregado para LER os dados de um programa.

c) A linha `WRITE(*,*) "Entre com o valor de A"` escreve o comentário que está entre aspas.

d) A linha `READ(*,*) A` lê um valor digitado dentro da janela DOS, aberta durante a execução do programa, e o atribui à variável A.

3) Executar **Build, Compile** para compilar o programa.

4) Gerar o programa-executável fazendo **Build, Build**.

Tabela 2.8 programa03.f90, versão B

```
INTEGER A, B, C, D, E, F, G

WRITE(*,*) "Entre com o valor de A"
READ(*,*) A

WRITE(*,*) "Entre com o valor de B"
READ(*,*) B

C = A + B
D = B - A
E = A * B
F = A / B
G = A ** B

WRITE(*,*) "A = ", A
WRITE(*,*) "B = ", B
WRITE(*,*) "C = A + B = ", C
WRITE(*,*) "D = B - A = ", D
WRITE(*,*) "E = A * B = ", E
WRITE(*,*) "F = A / B = ", F
WRITE(*,*) "G = A ** B = ", G

END
```

5) Ao se executar o programa, através de **Build, Execute**, surge uma janela do tipo DOS, mostrada na Figura 2.9, dentro da qual tem-se:

a) Na primeira linha, o comentário “Entre com o valor de A”, resultado do comando `WRITE(*,*) "Entre com o valor de A"` do programa.

b) Na segunda linha, o programa pára e fica aguardando que seja fornecido o valor da variável A, resultado do comando `READ(*,*) A` do programa. Para que o programa continue a sua execução é necessário **digitar o valor –6 para a variável A** e, em seguida, **clicar na tecla enter**.

c) Na terceira linha, o comentário “Entre com o valor de B”, resultado do comando `WRITE(*,*) "Entre com o valor de B"` do programa.

d) Na quarta linha, o programa pára e fica aguardando que seja fornecido o valor da variável B, resultado do comando `READ(*,*) B` do programa. Para que o programa continue a sua execução é necessário **digitar o valor 2 para a variável B** e, em seguida, **clicar na tecla enter**.

e) Em seguida, são apresentados os resultados mostrados na Figura 2.9, que devem ser iguais aos da Figura 2.7.

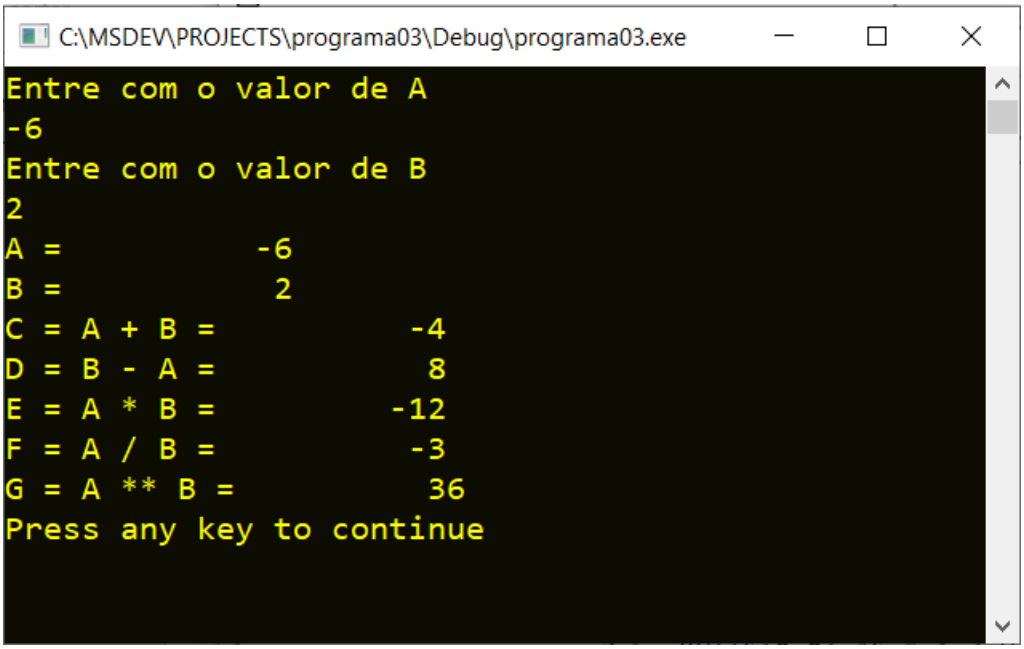

Figura 2.9 Resultado da execução do programa03.f90, versão B, com A = –6 e B = 2.

6) **Executar novamente o programa, entrando com os valores A = 1 e B = 2**. Em seguida, **analisar** cada resultado, comparando-o com o valor esperado obtido de um cálculo mental; eles devem ser iguais aos da Figura 2.8.

7) **Executar novamente o programa com A = 6 e B = 0**. Nenhum resultado é apresentado porque o programa não consegue dividir 6 por 0. Isso gera um erro que interrompe a execução normal do programa. Ao se implementar um programa, deve-se prepará-lo para que seja evitada qualquer divisão por zero. Isso pode ser feito através de comentários que informem ao usuário do programa, na janela da execução do programa ou no manual do programa, por exemplo, quais as variáveis que não podem ter valor nulo. Outra forma mais efetiva de evitar divisão por zero será vista em capítulo futuro.

8) O comando READ pode ser usado para interromper a execução do programa, em qualquer ponto do programa, com alguma finalidade. Para isso, basta usar o comando READ sem qualquer variável, ou seja, `READ(*,*)`. **Testar isso após os 5 cálculos do programa03.f90**. Para a execução do programa continuar, basta clicar na tecla ENTER, assim que se quiser.

2.8 EXERCÍCIOS

Exercício 2.1

Executar novamente o programa03.f90, versão B, com A = 2 e B = –1. Em seguida, analisar cada resultado, comparando-o com o valor esperado obtido de um cálculo mental, especialmente o caso da potenciação, cujo resultado deverá ser zero.

No Fortran, para abrir um projeto já existente, como o programa03, basta executar **File, Open Workspace**. Em seguida, **localizar a pasta do projeto** em Directories. Depois, deve-se **clicar 2 vezes sobre o nome do projeto**. Em seguida, **clicar sobre o nome do projeto com extensão "mdp"**, no caso programa03.mdp. Finalmente, deve-se **clicar no botão OK**.

Exercício 2.2

Em Fortran, números reais são representados com o sinal de ponto para separar a parte inteira da decimal; isto será visto com detalhes no Capítulo 3.

A tentativa de ler um número real, para atribuí-lo a uma variável do tipo inteiro, gera erro de execução do programa. Por exemplo, **executar novamente o programa03.f90, versão B, usando A = 1.5 e B = 0.4**.

Exercício 2.3

1) Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):
   a) definir como inteiras as variáveis VAR1, VAR2 e SOMA
   b) ler os valores das variáveis VAR1 e VAR2
   c) calcular a soma de VAR1 e VAR2 e atribuir o resultado à variável SOMA
   d) escrever os valores lidos de VAR1 e VAR2 juntamente com comentários para identificá-los, como na Figura 2.9
   e) escrever o resultado da variável SOMA juntamente com um comentário para identificá-lo
2) Compilar o programa-fonte
3) Gerar o programa-executável
4) Executar o programa para VAR1 = 3 e VAR2 = 4. O resultado esperado é SOMA = 7.

Exercício 2.4

1) Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):
   a) definir como inteiras as variáveis VAR1, VAR2 e X
   b) ler os valores das variáveis VAR1 e VAR2
   c) calcular o produto de VAR1 e VAR2 e atribuir o resultado à variável X
   d) escrever o seu nome completo
   e) escrever os valores lidos de VAR1 e VAR2 juntamente com comentários para identificá-los, como na Figura 2.9
   f) escrever o resultado da variável X juntamente com um comentário para identificá-lo
2) Compilar o programa-fonte
3) Gerar o programa-executável
4) Executar o programa para VAR1 = 3 e VAR2 = 4. O resultado esperado é X = 12.

Exercício 2.5

1) Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):
   a) definir como inteiras as variáveis VAR1, VAR2, P e D
   b) ler os valores das variáveis VAR1 e VAR2
   c) calcular a divisão de VAR1 por VAR2 e atribuir o resultado à variável D
   d) calcular VAR1 elevado à potência VAR2 e atribuir o resultado à variável P
   e) escrever o seu nome completo
   f) escrever os valores lidos de VAR1 e VAR2 juntamente com comentários para identificá-los, como na Figura 2.9
   g) escrever os resultados das variáveis D e P juntamente com comentários para identificá-los
2) Compilar o programa-fonte
3) Gerar o programa-executável
4) Executar o programa para VAR1 = 9 e VAR2 = 3. O resultado esperado é D = 3 e P = 729.

Exercício 2.6

1) Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):
   a) ler três números inteiros
   b) somar os três números e atribuir o resultado à variável SOMA
   c) calcular a média aritmética dos três números ao dividir SOMA por 3
   d) escrever os valores lidos, a SOMA e o valor da média aritmética juntamente com comentários para identificá-los, como na Figura 2.9
2) Compilar o programa-fonte
3) Gerar o programa-executável
4) Executar o programa com os valores 1, 2 e 3. Em seguida, analisar o resultado da média fornecido pelo programa comparando-o com o valor esperado obtido por um cálculo mental. Resultado esperado para a média = 2.
5) Repetir o item 4 para os valores 1, 1 e 2. Resultado esperado para a média = 1.
6) Repetir o item 4 para os valores 1, 1 e 0. Resultado esperado para a média = 0.

Exercício 2.7

1) Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):
   a) ler o primeiro valor (inteiro) de uma progressão aritmética (P.A.), denotado por A1
   b) ler a diferença (número inteiro) entre dois termos subsequentes da P.A., denotada por D
   c) ler o número (inteiro) de termos da P.A., denotado por N
   d) calcular o valor (inteiro) do último termo da P.A., denotado por AN, com AN = A1 + (N–1)*D
   e) calcular o valor (inteiro) da soma de todos os termos da P.A., denotado por SN, com a seguinte expressão: SN = (A1+AN)*N/2
   f) escrever os três valores lidos e os dois calculados juntamente com comentários para identificá-los, como na Figura 2.9
2) Compilar o programa-fonte
3) Gerar o programa-executável
4) Executar o programa para A1 = 1, D = 3 e N = 5. Os resultados devem ser AN = 13 e SN = 35.