# Capítulo 4. Variáveis do tipo CARACTER

**20 Mar 2025**

## OBJETIVOS DO CAPÍTULO

- Conceitos de: variáveis do tipo caracter, strings, funções intrínsecas
- Funções intrínsecas do FORTRAN para variáveis do tipo caracter: ADJUSTL, ADJUSTR, TRIM e LEN
- Operador de concatenação para variáveis do tipo caracter
- Novo comando do FORTRAN a usar: CHARACTER

## 4.1 programa4a.f90

Para inicializar as atividades deste capítulo, deve-se:

1) Executar o programa Fortran
2) No Fortran, **criar um projeto** com o nome **programa04**
3) No Fortran, **criar e inserir** no projeto o programa-fonte **programa4a.f90**
4) Dentro do espaço de edição do Fortran, conforme é mostrado na Figura 4.1, **copiar** exatamente o texto em vermelho mostrado na **Tabela 4.1**.

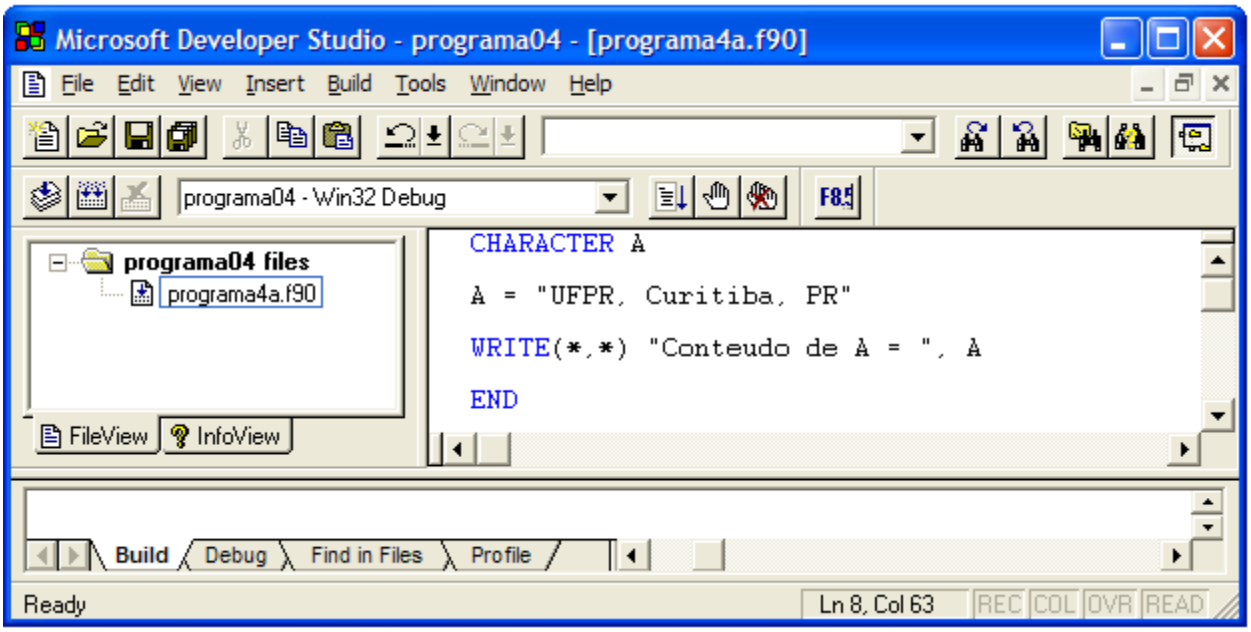


Figura 4.1 Programa4a.f90, versão 1.

5) Comentários sobre o programa:
   a) Nos capítulos 1 a 3 foram usados os comandos WRITE, END, INTEGER, READ e REAL da linguagem FORTRAN. No programa4a.f90 há um novo comando: CHARACTER. Ele é usado para definir variáveis do tipo caracter ou string, isto é, variáveis que podem guardar ou armazenar, na memória do computador,

comentários ou textos. Estes comentários ou textos podem incluir palavras, números, símbolos, espaços em branco ou frases. Alguns exemplos são (os conteúdos das variáveis estão separados por vírgula): UFPR, 5 de outubro de 2009, teste de hoje, TM-102, U-20/5%, L&L. As variáveis do tipo caracter também são chamadas de alfanuméricas porque envolvem letras e números.

b) No FORTRAN, o conteúdo ou string de uma variável do tipo caracter é definido entre aspas ou entre apóstrofos.

c) A linha `CHARACTER A` declara a variável A como sendo do tipo caracter. Este comando reserva um espaço na memória do computador, utilizando o nome ou rótulo A, para armazenar um único caracter alfanumérico.

d) A linha `A = "UFPR, Curitiba, PR"` atribui o comentário entre aspas à variável A. Este comentário tem 18 caracteres de informação.

Tabela 4.1 Programa4a.f90, versão 1.

```
CHARACTER A

A = "UFPR, Curitiba, PR"

WRITE(*,*) "Conteudo de A = ", A

END
```

6) Executar **Build, Compile** para compilar o programa
7) Gerar o programa-executável fazendo **Build, Build**
8) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 4.2, dentro da qual tem-se:

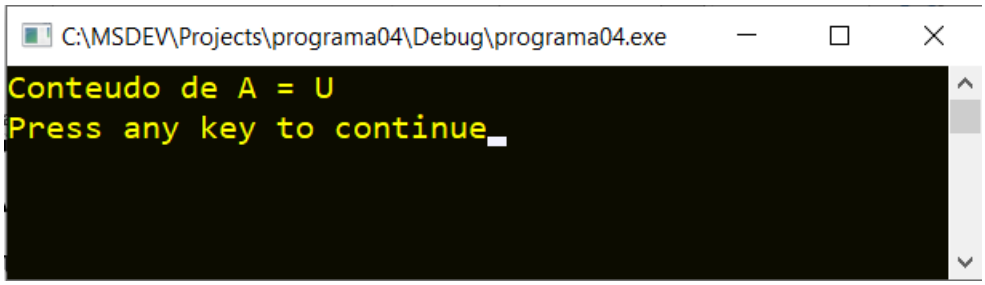


Figura 4.2 Resultado da execução do programa4a.f90, versão 1.

a) Na primeira linha, o comentário "Conteudo de A = ", e, em seguida, a letra U, resultado do comando `WRITE(*,*) "Conteudo de A = ", A` do programa.

b) Na segunda 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.

9) No FORTRAN, cada variável do tipo caracter deve ser declarada com a dimensão adequada à quantidade máxima de letras, números, espaços em branco e símbolos que ela poderá conter. Quando não se declara o tamanho, assume-se que seja apenas um caracter. Devido a isso, foi apresentada, como conteúdo da variável A, somente a letra U, de UFPR, na versão 1 do programa4a.f90, embora tenha sido atribuído à variável A um comentário com 18 caracteres.
10) Para declarar a dimensão de uma variável do tipo caracter, basta colocar o tamanho entre parênteses junto à palavra character, como exemplificado na versão 2 do programa4a.f90, na Tabela 4.2.

Tabela 4.2 Programa4a.f90, versão 2.

```
CHARACTER(50) A

A = "UFPR, Curitiba, PR"

WRITE(*,*) "Conteudo de A = ", A

END
```

11) **Alterar** a primeira linha do programa4a.f90 para ficar igual à Tabela 4.2.
12) **Compilar** novamente o programa
13) **Gerar** seu executável.
14) **Executar** o programa.
15) Agora, conforme a Figura 4.3, é apresentado o conteúdo completo da variável A, já que ele ocupa apenas 18 caracteres e a variável A foi dimensionada prevendo até 50 caracteres.

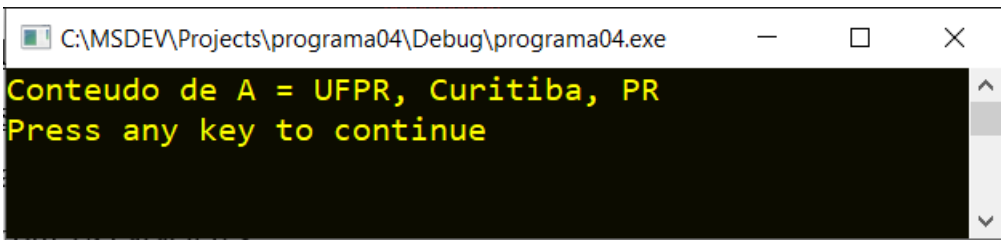


Figura 4.3 Resultado da execução do programa4a.f90, versão 2.

## 4.2 programa4b.f90

Nesta seção será usado o mesmo projeto da seção anterior. Portanto, deve-se executar o seguinte no Fortran:

1) **Clicar** sobre o nome do programa-fonte
2) **Edit, Cut** para retirar o programa-fonte do projeto.
3) **Criar e inserir** no projeto o programa-fonte **programa4b.f90**
4) Dentro do espaço de edição do Fortran, **copiar** exatamente o texto em vermelho mostrado na **Tabela 4.3**.

Tabela 4.3 Programa4b.f90.

```
CHARACTER(50) A
WRITE(*,*) ' Entre com o conteudo de "A" '
READ(*,*) A
WRITE(*,*) "Conteudo de 'A' = ", A
END
```

5) Comentários sobre o programa:
   a) A linha `CHARACTER(50) A` declara a variável A como sendo do tipo caracter. Este comando reserva um espaço na memória do computador, utilizando o nome ou rótulo A, para armazenar até 50 caracteres alfanuméricos.
   b) Na linha do primeiro WRITE, estão sendo usados apóstrofos para delimitar o comentário, e aspas para ressaltar o nome da variável. O mesmo é feito no segundo WRITE, mas invertendo aspas com apóstrofos.
   c) O comando READ do FORTRAN também pode ser usado para atribuir "valor" ou conteúdo a uma variável do tipo caracter, desde que o conteúdo esteja entre aspas ou apóstrofos. Um exemplo é mostrado na linha `READ(*,*) A`.
6) Executar **Build, Compile** para compilar o programa
7) Gerar o programa-executável fazendo **Build, Build**
8) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 4.4, dentro da qual tem-se:

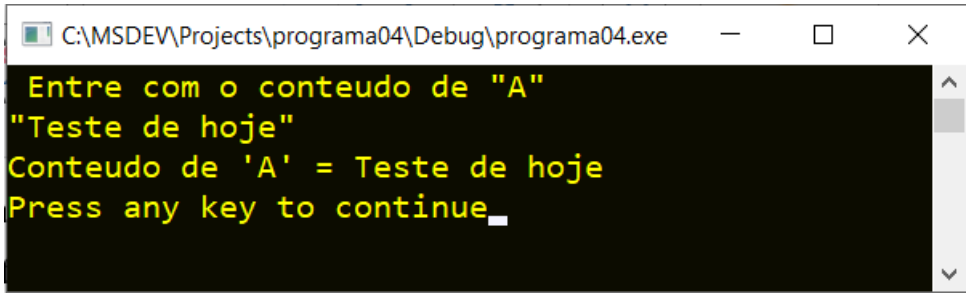


Figura 4.4 Resultado da execução do programa4b.f90.

   a) Na primeira linha, o comentário ' Entre com o conteudo de "A" ', resultado do comando `WRITE(*,*) ' Entre com o conteudo de "A" '` do programa.
   b) Na segunda linha, o programa pára e fica aguardando que seja fornecido o conteúdo da variável A, resultado do comando `READ(*,*) A` do programa. Para que o programa continue sua execução é necessário **digitar, entre aspas ou apóstrofos, um conteúdo para a variável A**, por exemplo, "**Teste de hoje**" e, em seguida, **clicar na tecla enter**.
   c) Na terceira linha, o conteúdo da variável A.

9) Executar novamente o programa, através de **Build, Execute**, testando outros conteúdos para a variável A e vendo o resultado. Testar, por exemplo, **uma frase com pelo menos duas palavras mas sem usar aspas** para definir o conteúdo de A.

## 4.3 programa4c.f90

1) Nesta seção será usado o mesmo projeto da seção anterior. Portanto, deve-se executar o seguinte no Fortran:
   a) **Clicar** sobre o nome do programa-fonte
   b) **Edit, Cut** para retirar o programa-fonte do projeto.
2) No Fortran, **criar e inserir** no projeto o programa-fonte **programa4c.f90**
3) Dentro do espaço de edição do Fortran, **copiar** exatamente o texto em vermelho mostrado na **Tabela 4.4**.

Tabela 4.4 Programa4c.f90.

```
CHARACTER(25) LOCAL, DIA
CHARACTER(10) HORA
CHARACTER(90) FRASE

WRITE(*,*) "Entre com o local"
READ(*,*) LOCAL

WRITE(*,*) "Entre com o dia"
READ(*,*) DIA

WRITE(*,*) "Entre com a hora"
READ(*,*) HORA

FRASE = LOCAL // ", " // DIA // ", as " // HORA // " horas"

WRITE(*,*) "Conteudo de FRASE =", FRASE

END
```

4) Comentários sobre o programa:
   a) A linha `CHARACTER(25) LOCAL, DIA` declara duas variáveis, LOCAL e DIA, como sendo do tipo caracter, cada uma podendo armazenar até 25 caracteres alfanuméricos.
   b) A linha `CHARACTER(10) HORA` declara a variável HORA como sendo do tipo caracter, podendo armazenar até 10 caracteres alfanuméricos.
   c) A linha `CHARACTER(90) FRASE` declara a variável FRASE como sendo do tipo caracter, podendo armazenar até 90 caracteres alfanuméricos.

d) A linha `FRASE = LOCAL // ", " // DIA // ", as " // HORA // " horas"` define o conteúdo da variável FRASE com base em alguns comentários que estão entre aspas e no conteúdo das variáveis LOCAL, DIA e HORA. Isso é possível devido ao uso do operador de concatenação cujo símbolo é composto por duas barras (//). Este operador permite juntar duas ou mais variáveis do tipo caracter e comentários.

e) O operador (//) deve sempre estar entre duas variáveis, dois comentários ou entre uma variável e um comentário, nunca iniciando ou finalizando a linha de comando.

5) Executar **Build, Compile** para compilar o programa
6) Gerar o programa-executável fazendo **Build, Build**
7) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 4.5, dentro da qual tem-se:

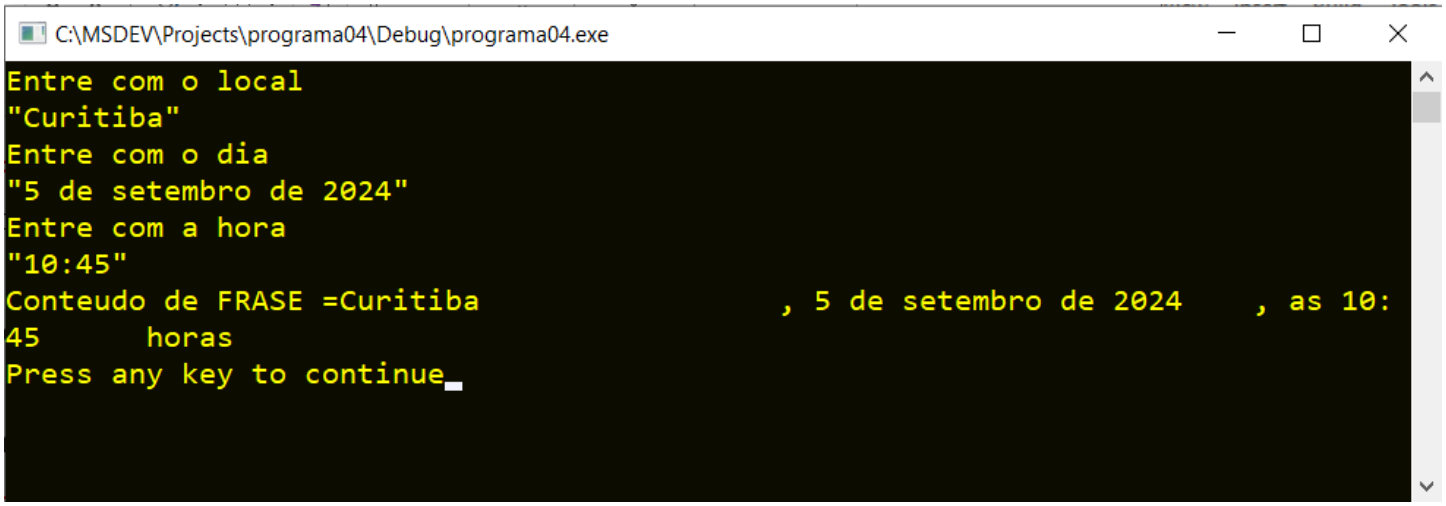


Figura 4.5 Resultado da execução do programa4c.f90.

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

b) Na segunda linha, o programa pára e fica aguardando que seja fornecido o conteúdo da variável LOCAL, resultado do comando `READ(*,*) LOCAL` do programa. Para que o programa continue sua execução é necessário **digitar o conteúdo da variável LOCAL**, por exemplo, “**Curitiba**” e, em seguida, **clicar na tecla enter**.

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

d) Na quarta linha, o programa pára e fica aguardando que seja fornecido o conteúdo da variável DIA, resultado do comando `READ(*,*) DIA` do programa. Para que o programa continue sua execução é necessário **digitar o conteúdo da variável DIA**, por exemplo, “**5 de setembro de 2024**” e, em seguida, **clicar na tecla enter**.

e) Na quinta linha, o comentário “Entre com a hora”, resultado do comando `WRITE(*,*) "Entre com a hora"` do programa.

f) Na sexta linha, o programa pára e fica aguardando que seja fornecido o conteúdo da variável HORA, resultado do comando `READ(*,*) HORA` do programa. Para que o programa continue sua execução é necessário **digitar o conteúdo da variável HORA**, por exemplo, "10:45" e, em seguida, **clicar na tecla enter**.

g) Na sétima linha, o comentário "Conteúdo de FRASE =" e, em seguida, o conteúdo da variável FRASE, resultado do comando `WRITE(*,*) "Conteudo de FRASE =", FRASE` do programa e da linha anterior onde se definiu a variável FRASE.

8) Deve-se notar na Figura 4.5 que a escrita do conteúdo da variável FRASE ocupou mais de uma linha. Isso ocorre devido a sua extensão ser muito grande, isto é, superior ao número disponível de colunas na janela do tipo DOS.

9) No FORTRAN, se o conteúdo de uma variável é definido com menos caracteres do que o máximo declarado, a diferença entre os caracteres ocupados e o máximo previsto é preenchida com espaços em branco, como no conteúdo da variável FRASE. Por exemplo, a variável LOCAL foi declarada com tamanho máximo de 25 caracteres. Mas seu conteúdo foi definido com "Curitiba", palavra que ocupa apenas 8 caracteres. Neste caso, os demais 17 caracteres são preenchidos com espaços em branco. O mesmo problema ocorre com as demais variáveis. Formas de se resolver este problema são abordadas na próxima seção.

10) **Executar** novamente o programa com outros conteúdos para as variáveis LOCAL, DIA e HORA.

## 4.4 programa4d.f90

1) Nesta seção será usado o mesmo projeto da seção anterior deste capítulo. Portanto, deve-se executar o seguinte no Fortran:
   a) **Clicar** sobre o nome do programa-fonte
   b) **Edit, Cut** para retirar o programa-fonte do projeto.
2) No Fortran, **criar e inserir** no projeto o programa-fonte **programa4d.f90**
3) Dentro do espaço de edição do Fortran, **copiar** exatamente o texto em vermelho mostrado na **Tabela 4.5**.
4) Comentários sobre o programa:
   a) A linha `CHARACTER(9) A` declara a variável A como sendo do tipo caracter, podendo armazenar até 9 caracteres alfanuméricos.
   b) A linha `INTEGER L` declara a variável L como sendo do tipo inteiro.
   c) Conforme pode-se ver na **Figura 4.6**, ADJUSTL, ADJUSTR, TRIM e LEN estão escritos em azul dentro do Fortran. Elas são chamadas de funções intrínsecas do FORTRAN, ou seja, são funções ou comandos que existem dentro da linguagem FORTRAN. Estas funções são usadas para manipular strings, isto é, conteúdos de variáveis do tipo caracter.
   d) A lista completa das funções para variáveis do tipo caracter pode ser vista no manual do Fortran. Para acessá-lo, dentro da subjanela do lado esquerdo, deve-se executar: clicar uma vez sobre o símbolo **?InfoView**; e acessar as opções **Reference, Procedures, Character Procedures**.

e) A linha `WRITE(*,*) "A =(", A, ")"` escreve em sequência o conteúdo de três strings que estão separadas por vírgula. A primeira é o comentário `A =(`; a segunda, o conteúdo da variável A; e a terceira, `)`. As quatro linhas seguintes do programa fazem o mesmo, isto é, cada uma escreve três strings em seqüência. Mas, nestes casos, em vez de escrever o conteúdo da variável A, escreve-se o resultado de alguma função operando sobre A, conforme as explicações a seguir.

Tabela 4.5 Programa4d.f90.

```
CHARACTER(9) A
INTEGER L

WRITE(*,*) "Entre com o conteudo de A ate 9 caracteres"
WRITE(*,*) "0123456789"
READ(*,*) A

WRITE(*,*) "                    0123456789"
WRITE(*,*) "A                  =(", A, ")"
WRITE(*,*) "Com ADJUSTL        =(", ADJUSTL(A), ")"
WRITE(*,*) "Com ADJUSTR        =(", ADJUSTR(A), ")"
WRITE(*,*) "Com TRIM           =(", TRIM(A), ")"
WRITE(*,*) "Com ADJUSTL e TRIM =(", TRIM(ADJUSTL(A)), ")"

L = LEN(A)
WRITE(*,*) "L de A = ", L

L = LEN(TRIM(ADJUSTL(A)))
WRITE(*,*) "L de A com TRIM e ADJUSTL = ", L

L = LEN("Teste")
WRITE(*,*) 'L de "Teste" = ', L

END
```

f) A função ADJUSTL transfere os espaços em branco que estão à esquerda de uma string e os passa à direita dela.

g) A função ADJUSTR transfere os espaços em branco que estão à direita de uma string e os passa à esquerda dela.

h) A função TRIM elimina os espaços em branco que estão à direita de uma string.

i) O resultado das funções ADJUSTL, ADJUSTR e TRIM são strings.

j) A função LEN conta o número de caracteres de uma string ou variável do tipo caracter. Seu resultado é um número inteiro.

k) Na instrução `TRIM(ADJUSTL(A))` do programa, a função TRIM opera sobre o resultado da função ADJUSTL sobre a variável A.

l) Na linha `L = LEN(A)` do programa, a função LEN opera sobre a variável A e atribui o resultado à variável inteira L.

m) Na linha `L = LEN(TRIM(ADJUSTL(A)))` do programa, a função LEN opera sobre o resultado da função TRIM, que opera sobre o resultado da função ADJUSTL sobre a variável A. Finalmente, o resultado da função LEN é atribuído à variável inteira L.

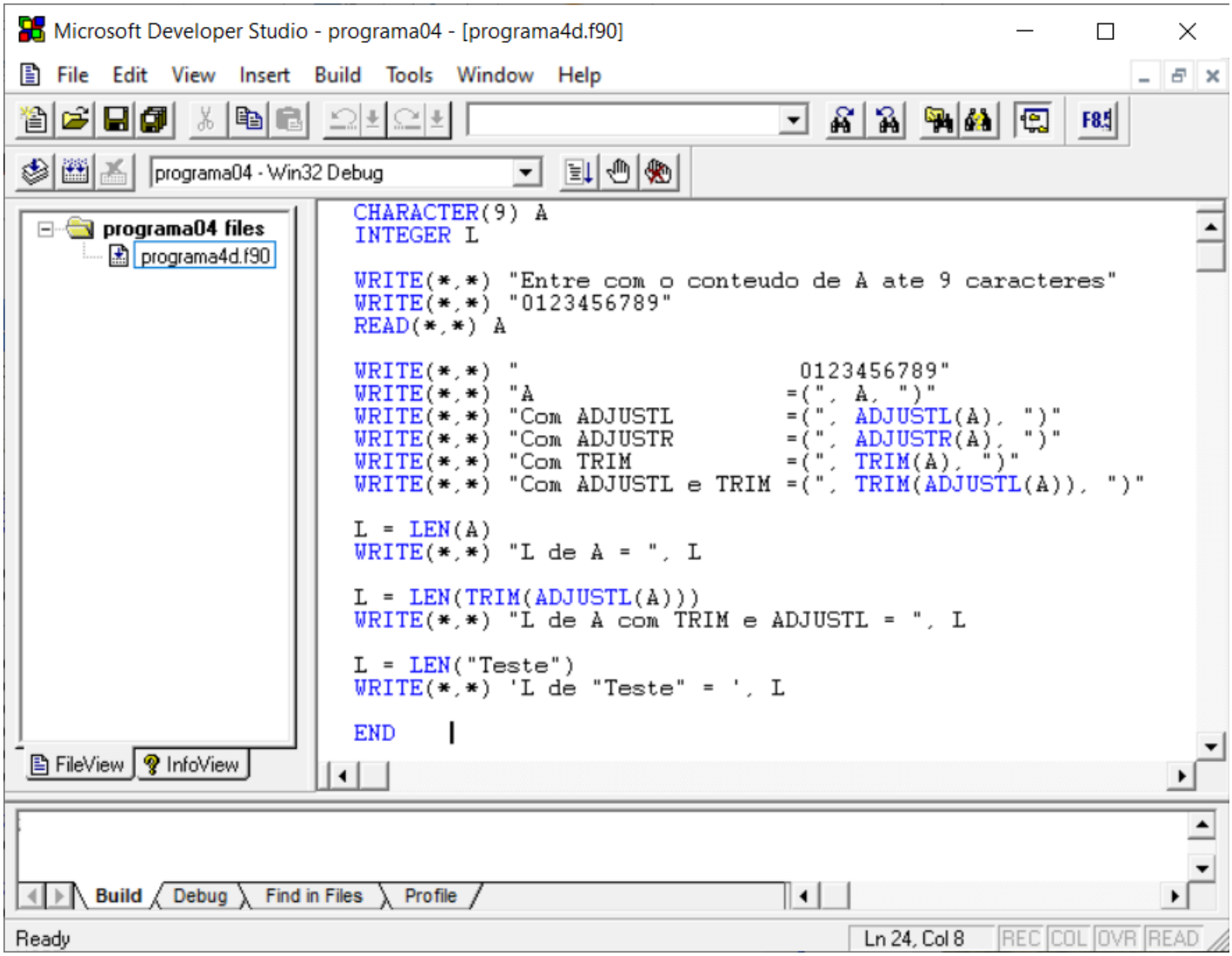


Figura 4.6 Programa4d.f90.

5) Executar **Build, Compile** para compilar o programa

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

7) Ao se executar o programa, através de **Build, Execute**, surge uma janela, mostrada na Figura 4.7, dentro da qual tem-se:

a) Na primeira linha, o comentário “Entre com o conteudo de A ate 9 caracteres”, resultado do comando `WRITE(*,*) "Entre com o conteudo de A ate 9 caracteres"` do programa.

b) Na segunda linha, o comentário 0123456789, resultado do comando `WRITE(*,*) "0123456789"` do programa. Este comentário é uma espécie de régua para se perceber claramente o número de caracteres usados para definir a variável A, e quantos espaços em branco existem antes da string de A e depois dela. O zero é usado porque a definição da variável A exige o sinal de aspas, que não conta em seu conteúdo.

c) Na terceira linha, o programa pára e fica aguardando que seja fornecido o conteúdo da variável A, resultado do comando `READ(*,*) A` do programa. Para que o programa continue sua execução é necessário **digitar o conteúdo da variável A**, por exemplo, " **Rio 45**" e, em seguida, **clicar na tecla enter**. Deve-se perceber que há um espaço em branco antes da palavra Rio.

d) Na quarta linha, 20 espaços em branco seguidos pelo comentário "0123456789", resultado do comando `WRITE(*,*) " 0123456789"` do programa. Este comentário é uma espécie de régua para se perceber claramente os resultados a seguir.

e) Nas linhas seguintes, são apresentados os resultados do conteúdo da variável A e das funções ADJUSTL, ADJUSTR, TRIM e LEN operando sobre a variável A. Os parênteses marcam o início e o fim de cada string.

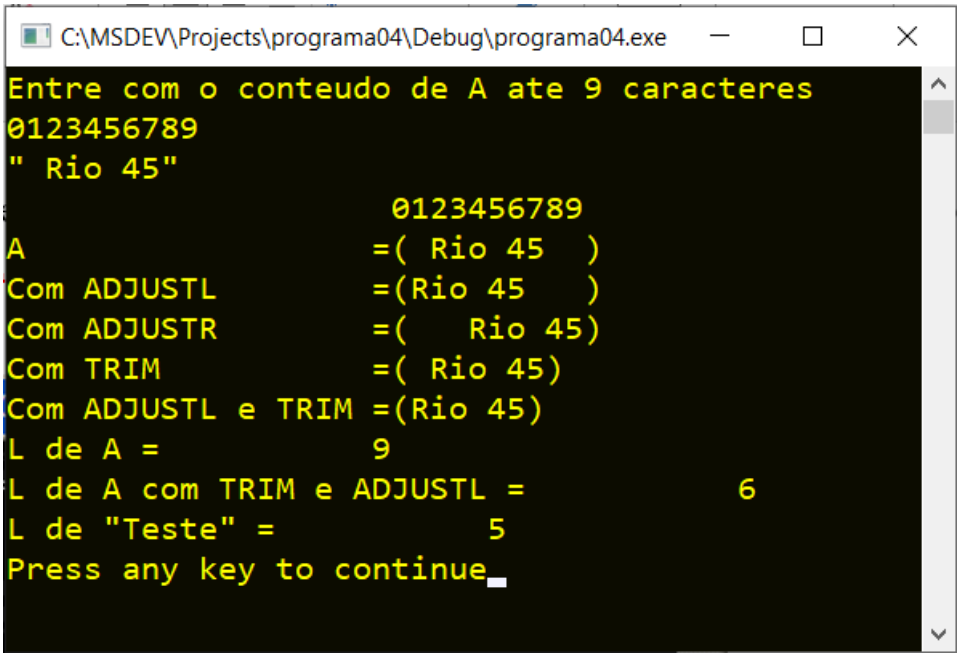


Figura 4.7 Resultado da execução do programa4d.f90.

8) **Analisar** cada resultado mostrado na Figura 4.7, considerando o conteúdo da variável A e as definições apresentadas acima, nesta seção, para as funções ADJUSTL, ADJUSTR, TRIM e LEN.

9) **Executar** novamente o programa com outros conteúdos para a variável A; por exemplo: "Curitiba", "Florianopolis", e "Rio" a partir da coluna 7. **Analisar** os resultados de cada nova execução do programa.

## 4.5 EXERCÍCIOS

### Exercício 4.1

1) Editar um programa-fonte em FORTRAN para executar o seguinte algoritmo (passos):
   a) declarar quatro variáveis do tipo caracter com dimensões de 5, 10, 20 e 30 caracteres
   b) ler o conteúdo das quatro variáveis
   c) escrever o conteúdo de cada uma das quatro variáveis
   d) usando o operador de concatenação, escrever combinações das quatro variáveis lidas
   e) aplicar e escrever o resultado da função ADJUSTL a cada uma das quatro variáveis lidas
   f) aplicar e escrever o resultado da função ADJUSTR a cada uma das quatro variáveis lidas
   g) aplicar e escrever o resultado da função TRIM a cada uma das quatro variáveis lidas
   h) aplicar e escrever o resultado da função TRIM(ADJUSTL) a cada uma das quatro variáveis lidas
   i) aplicar e escrever o resultado da função LEN a cada uma das quatro variáveis lidas
   j) aplicar e escrever o resultado da função LEN(TRIM(ADJUSTL)) a cada uma das quatro variáveis lidas
2) Compilar o programa-fonte
3) Gerar o programa-executável
4) Executar o programa. Em seguida, comparar os resultados escritos com o esperado para cada caso.

### Exercício 4.2

1) Definir X como uma variável inteira
2) Definir C1, C2 e C3 como variáveis do tipo caracter, sendo C1 e C2 com dimensão 20 e C3 com dimensão 50
3) Ler os conteúdos de C1 e C2
4) Concatenar C1 com C2, incluindo um espaço em branco entre eles, e atribuir o resultado a C3.
5) Calcular o número de caracteres que resulta da aplicação simultânea das funções TRIM e ADJUSTL a C3; atribuir o resultado à variável X.
6) Escrever o nome completo do aluno.
7) Escrever os resultados de C3 e X junto com seus nomes, sendo um em cada linha
8) Executar o programa com C1 = “aula 4” e C2 = “de Fortran”. Os resultados esperados são:

```
C3=aula 4                de Fortran
X=          31
```

**Exercício 4.3**

1) Definir X como uma variável inteira
2) Definir C1, C2 e C3 como variáveis do tipo caracter, sendo C1 e C2 com dimensão 20 e C3 com dimensão 50
3) Ler os conteúdos de C1 e C2
4) Concatenar as variáveis C1 e C2 com a aplicação simultânea das funções TRIM e ADJUSTL a cada uma destas duas variáveis; incluir " de " entre elas, e atribuir o resultado a C3.
5) Calcular o número de caracteres que resulta da aplicação simultânea das funções TRIM e ADJUSTL a C3; atribuir o resultado à variável X.
6) Escrever o nome completo do aluno.
7) Escrever os resultados de C3 e X junto com seus nomes, sendo um em cada linha
8) Executar o programa com C1 = “aula 5” e C2 = “Fortran”. Os resultados esperados são:

```
C3=aula 5 de Fortran
X=         17
```