Programação .NET

Professor Ricardo Frohlich da Silva

Programação .NET

- Unidade 1 - Introdução a programação C#
  - 1.1 Introdução ao C#: Conceitos, Ambiente de Desenvolvimento e Primeiro Programa
  - 1.2 Controle de Fluxo e Métodos: Estruturas Condicionais, Repetição e Funções
- Unidade 2 - Programação Orientada a Objetos com C#
  - 2.1 Fundamentos de POO: Classes, Objetos, Herança e Encapsulamento
  - 2.2 Aplicações de POO: Polimorfismo, Abstração e Relacionamentos entre Classes

Programação .NET

- Unidade 3 - Construindo Aplicações com .NET Framework
  - 3.1 Estrutura do .NET Framework
  - 3.2 Criando Aplicações Básicas com .NET
- Unidade 4 - Explorando Frameworks
  - 4.1 Introdução ao ASP.NET Core
  - 4.2 Entity Framework

Programação .NET

- Abertura da disciplina
  - Você já teve contato com lógica de programação?
  - Já ouviu falar sobre Programação Orientada a Objetos?
  - Esta disciplina fornecerá bases sólidas para o desenvolvimento de aplicações utilizando C#, introduzindo conceitos fundamentais e boas práticas para a construção de software eficiente e escalável
  - Da programação básica ao desenvolvimento de uma aplicação completa, utilizando o .NET Framework

Programação .NET

- Caracterização da metodologia de ensino

  - A disciplina será conduzida por meio de aulas expositivas e práticas, promovendo a interação e o debate sobre os conteúdos. Os alunos realizarão exercícios teóricos e práticos, atividades em laboratório e desafios baseados em problemas reais.

  - A plataforma Minha UFN será utilizada como suporte para materiais e entregas de atividades. Além disso, serão aplicadas metodologias ativas, como aprendizagem baseada em projetos.

Programação .NET

- Avaliação da aprendizagem
  - O processo de avaliação da disciplina será quantitativo, composto por três notas resultantes dos produtos de aprendizagem e dos exercícios realizados em sala de aula e laboratório. As notas 1, 2 e 3 serão obtidas da seguinte forma:
    - N1 = (Produto de aprendizagem 1 * 0.6) + (Exercícios * 0.4)
    - N2 = (Produto de aprendizagem 2 * 0.6) + (Exercícios e trabalhos * 0.4)
    - N3 = (Produto de aprendizagem 3 * 0.6) + (Exercícios e trabalhos * 0.4)
  - A nota final da disciplina será obtida por meio da média aritmética simples entre as notas N1, N2 e N3 conforme cálculo abaixo:
    - Nota Final = (N1 + N2 + N3) /3O critério de arredondamento da nota final levará em conta a participação dos alunos em sala de aula e a entrega dos exercícios. Serão considerados aprovados na disciplina os alunos que obtiverem a nota final igual ou superior a sete (6,0) e frequência igual ou superior a setenta e cinco por cento(75%).
    - Aluno Aprovado = (Nota Final >= 6,0) + (Frequência >= 75%)..

Programação .NET

- Planejamento das aulas e das atividades
  - ACOMPANHAR SEMANALMENTE O PLANO DE ENSINO E APRENDIZAGEM!

Programação .NET

- C# e .NET são duas tecnologias intimamente relacionadas, mas distintas. Aqui está uma explicação sobre a relação entre elas:
  - C#:
    - C# (C Sharp) é uma linguagem de programação moderna, orientada a objetos e de propósito geral, desenvolvida pela Microsoft.
    - Ela foi projetada para ser simples, eficiente, segura e de alto desempenho.
    - C# é amplamente utilizada para desenvolver uma variedade de aplicativos, desde aplicativos de desktop até aplicativos da web e móveis.

Programação .NET

- .NET:
  - .NET é uma estrutura de software desenvolvida pela Microsoft que fornece um ambiente de execução para executar e executar aplicativos.
  - Ele consiste em uma grande biblioteca de classes (Framework Class Library - FCL) e um ambiente de execução comum, conhecido como Common Language Runtime (CLR).
  - O .NET oferece suporte a várias linguagens de programação, incluindo C#, Visual Basic .NET, F# e outras.
  - Ele é projetado para ser multiplataforma e é amplamente utilizado para desenvolver uma variedade de aplicativos, desde aplicativos de desktop do Windows até aplicativos da web e móveis.

Programação .NET

- C# é uma das linguagens de programação suportadas pela plataforma .NET.
- Quando você escreve um código em C#, ele é compilado em bytecode intermediário (IL - Intermediate Language) que é executado pelo CLR.
- O CLR é responsável por gerenciar a execução do código, incluindo a coleta de lixo, a segurança, o controle de tipo, entre outros.
- A biblioteca de classes .NET (FCL) fornece um conjunto rico de funcionalidades para desenvolver aplicativos, incluindo manipulação de arquivos, comunicação de rede, acesso a banco de dados, entre outros.
- Portanto, ao desenvolver em C#, você está escrevendo código que é executado no ambiente .NET, aproveitando as funcionalidades fornecidas pela estrutura .NET.

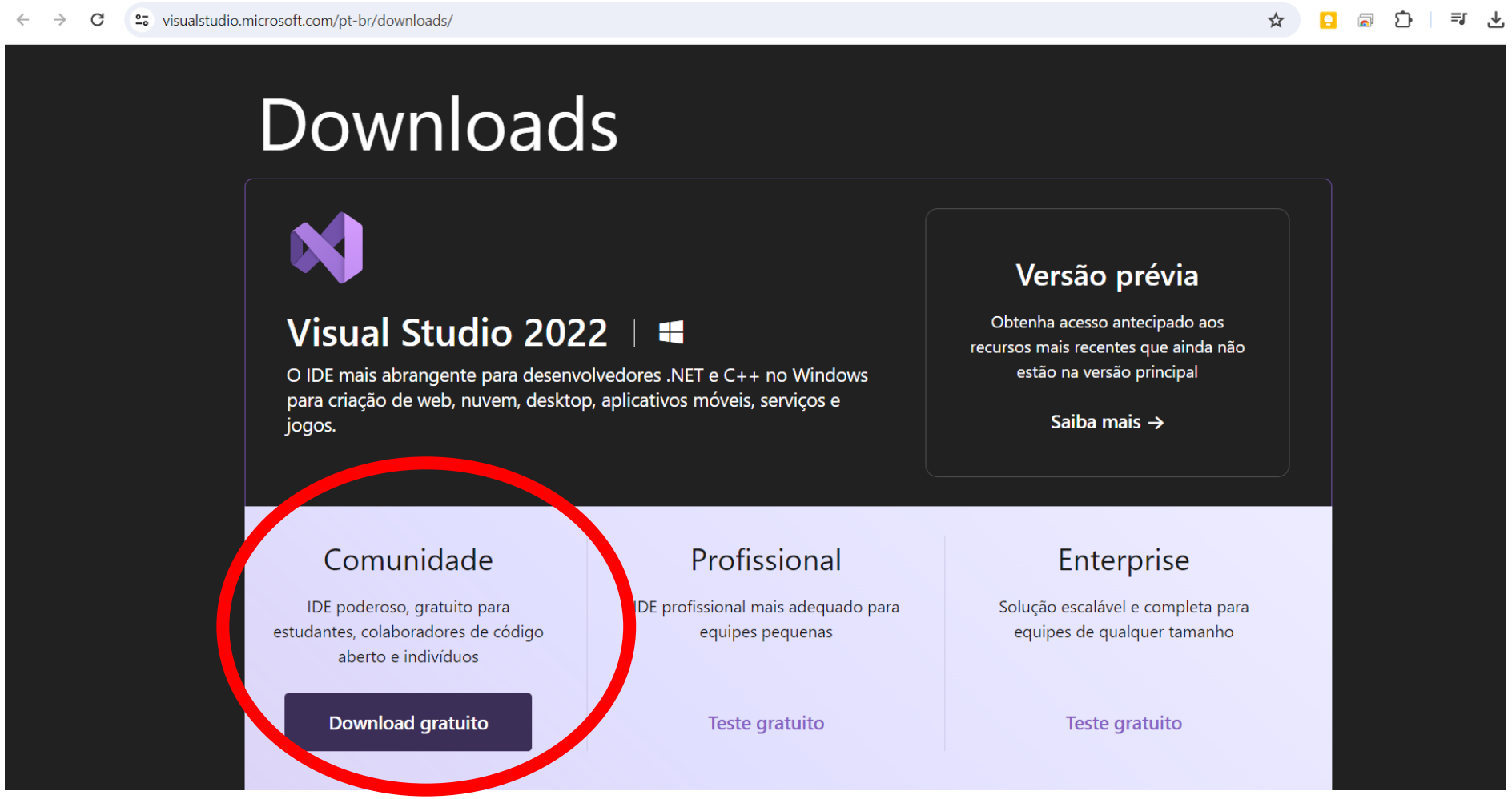
Visual Studio

- O Visual Studio é um ambiente de desenvolvimento integrado (IDE) desenvolvido pela Microsoft.

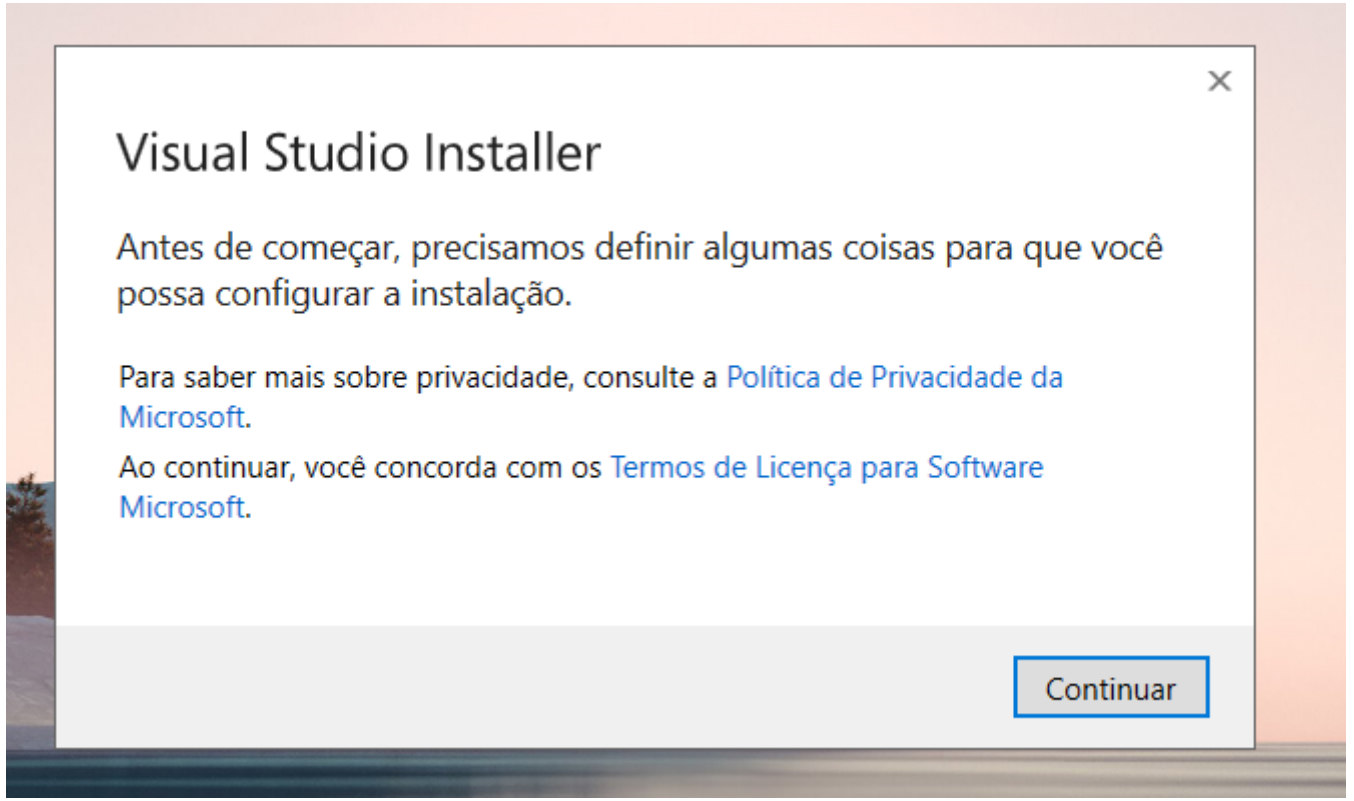
- Ele oferece uma ampla gama de ferramentas e recursos para desenvolvedores criarem aplicativos para diversas plataformas, incluindo aplicativos de desktop, web, móveis e em nuvem.

- https://visualstudio.microsoft.com/pt-br/downloads/

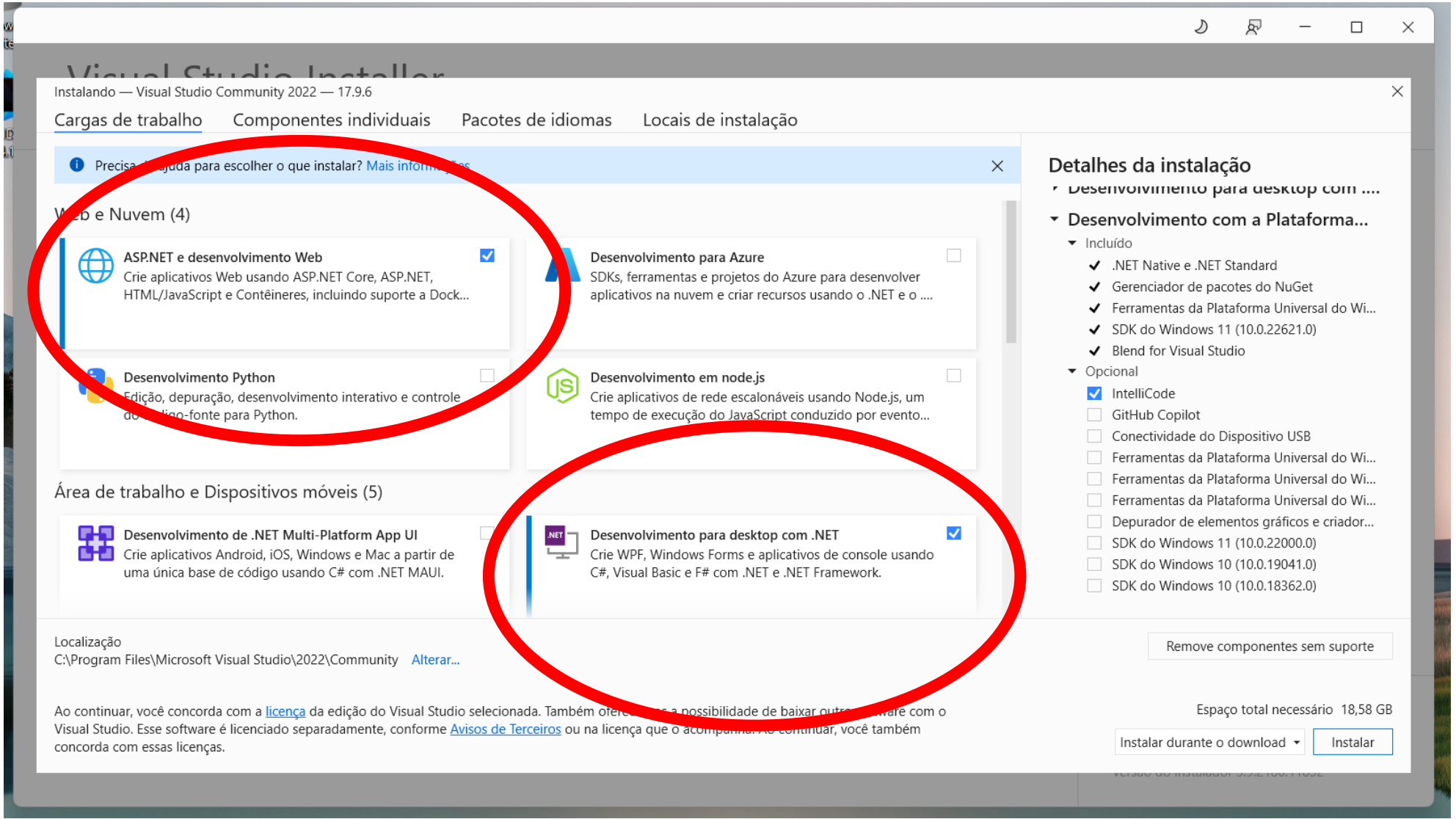
Visual Studio



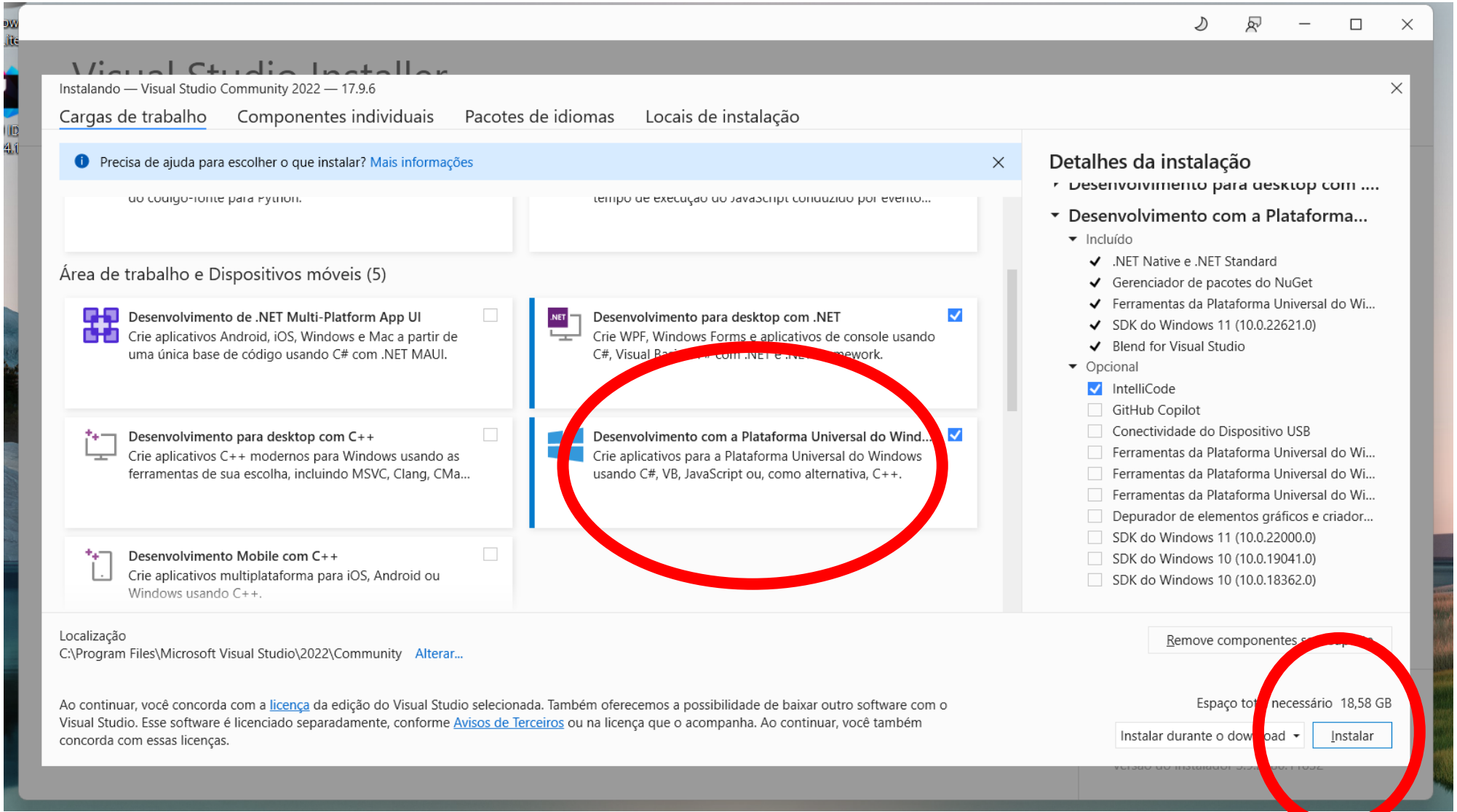
Visual Studio



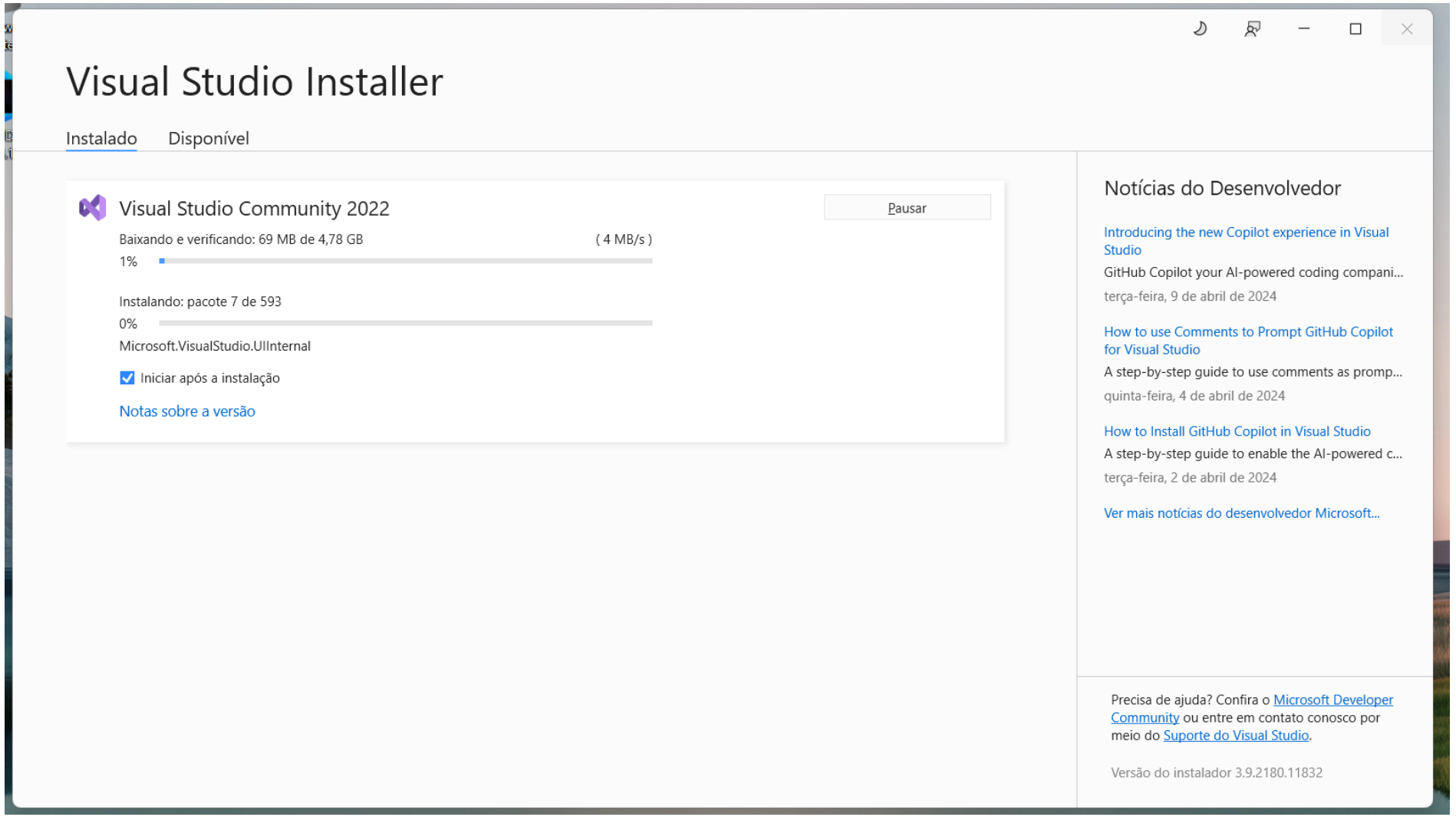
Visual Studio



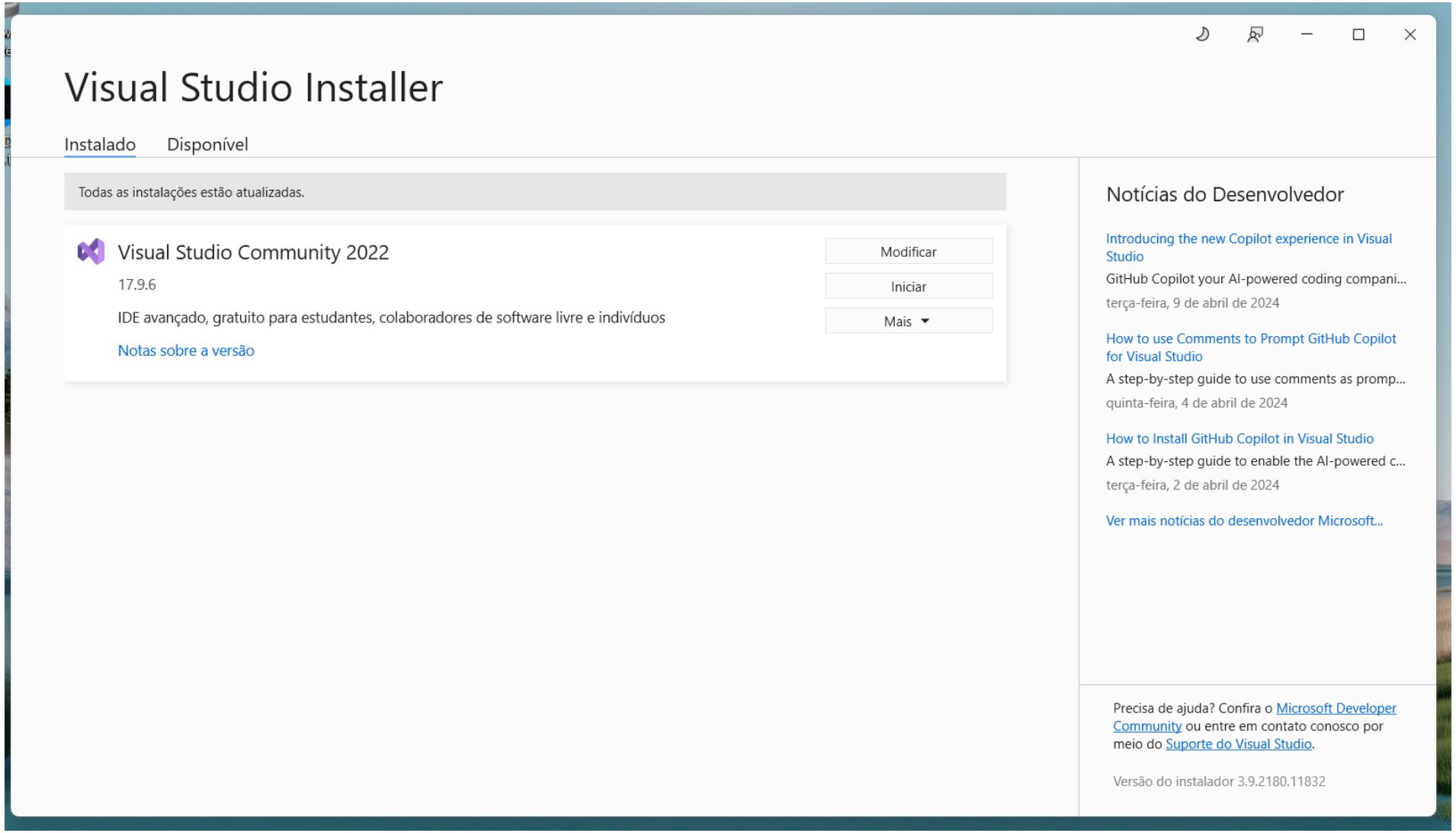
Visual Studio



Visual Studio



Visual Studio



Comandos de entrada e saída

- Para permitir a interação do usuário com os programas, existem os comandos que fazem a entrada (leitura) e saída (escrita) de dados.

- O dispositivo padrão de entrada é o teclado e o dispositivo padrão de saída de dados é o monitor de vídeo.

Comando de saída

- **Console.WriteLine** (`string de controle`)

  - Na string de controle iremos colocar os dados que serão impressos e, se necessário, o formato que estes serão exibidos.

Comando de saída

- Exemplo 1: mostrar um texto na tela:

```
static void Main(string[] args)
{
    Console.WriteLine(“Eita mundão sô!");
}
```

- Quando queremos mostrar somente um texto fixo, sem variáveis, basta colocar o texto entre aspas duplas

Comando de saída

- Exemplo 2:

```
 7            int n1, n2;
 8            n1 = 5;
 9            n2 = 3;
10            Console.WriteLine("N1 é igual a "+n1);
11            Console.WriteLine("N2 é igual a "+n2);
12            int soma;
13            soma = n1 + n2;
14            Console.WriteLine("A soma das variaveis é igual a "+soma);
```

Comando de saída

- Exemplo 3: Trabalhando com o tipo DateTime:

```
7           var dt = new DateTime();
8           Console.WriteLine(dt);
9
10          Console.WriteLine("Dia = "+dt.Day);
11          Console.WriteLine("Mês = "+dt.Month);
12          Console.WriteLine("Ano = "+dt.Year);
13          Console.WriteLine("Hora = "+dt.Hour);
14          Console.WriteLine("Minuto ="+dt.Minute);
15          Console.WriteLine("Pegando o DateTime de agora: ");
16          dt = DateTime.Now;
17          Console.WriteLine("Dia = " + dt.Day);
18          Console.WriteLine("Mês = " + dt.Month);
19          Console.WriteLine("Ano = " + dt.Year);
20          Console.WriteLine("Hora = " + dt.Hour);
21          Console.WriteLine("Minuto = " + dt.Minute);
```

Tipos de dados da Linguagem C#

- Tipos básicos de dados
  - char (caractere).
    - Ex: `char letra1 = 'A';`
  - int (inteiro).
    - Ex: `int valor1 = 54;`
  - float (ponto flutuante).
    - Ex: `float valor2 = 43.6778;`
  - double (ponto flutuante de precisão dupla).
    - Ex: `double valor3 = 32.45345346;`

Variáveis na Linguagem C#

- Variável é uma posição nomeada de memória, que possui um nome para identificação e corresponde a um tipo de dado.

- Declaração de variáveis na linguagem C#:
  - `TIPO_DE_DADO nome_var1, nome_var2;`

- É possível na declaração definir um valor inicial da variável.

Variáveis na Linguagem C#

- Exemplos de declaração:

```
int x, y, z=10;
float a = 3, c = 4.23, d;
char i, j = 'a';
```

Operadores na Linguagem C#

- Operadores são símbolos utilizados para realizar operações lógicas e aritméticas sobre operandos.
- Operadores aritméticos atuam sobre variáveis, constantes e funções numéricas e produzem um resultado numérico.
- Operadores possuem prioridades em relação aos outros, ou seja, qual operação será executada primeiro em relação as demais.

Operadores na Linguagem C#

- A tabela abaixo apresenta os operadores da linguagem C# e suas prioridades:

Prioridade	Operador	Operação
1	-	Inversão de sinal
2	*	Multiplicação
2	/	Divisão
2	%	Resto de Divisão
3	+	Adição
3	-	Subtração

Operadores na Linguagem C#

- Os operadores que possuem mesma prioridade são executados na ordem em que aparecem quando a expressão é lida da esquerda para a direita.

- O uso dos parênteses define uma ordem de execução prioritária em relação à prioridade dos operadores da linguagem.

- Exemplos:
  - `x + 5 * y - 4`
    - a primeira operação realizada é a multiplicação, em seguida é executada a soma e por fim a subtração.
  - `(x+5) * (y-4)`
    - para estabelecer uma precedência diferente, podem ser utilizados os parênteses.
    - Neste caso, primeiro é realizado a soma, depois a diferença, e por fim a multiplicação.

Exemplo

```
16          double nota1, nota2, nota3, media;
17          Console.WriteLine("Atribuindo três notas para um aluno: ");
18          nota1 = 7.3;
19          Console.WriteLine("Nota 1 é igual a "+nota1);
20          nota2 = 6.4;
21          Console.WriteLine("Nota 2 é igual a "+nota2);
22          nota3 = 4.7;
23          Console.WriteLine("Nota 3 é igual a "+nota3);
24
25          Console.WriteLine("Calculando a média do aluno: ");
26          media = nota1 + nota2 + nota3 / 3;
27          Console.WriteLine("Media errada do aluno causada pela falta de parenteses = "+media);
28
29          media = (nota1 + nota2 + nota3) / 3;
30          Console.WriteLine("Media correta do aluno = " + media);
```

Operadores de incremento e decremento

- Muitas vezes será necessário realizar o incremento (somar determinado valor) ou decremento (subtrair determinado valor) em algumas variáveis, por exemplo:

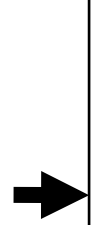
```
static void Main(string[] args)
{
  int a=5, b=10;
  a = a-1;  //a = 4
  b = b+1;  //b = 11
  a = a+5;  //a = 9
  b = b-3;  //b = 8
 }
```

Obs: isto é um comentário!!

Operadores de incremento e decremento

- A maioria das linguagens possibilita realizar este tipo de operação de uma forma mais simples:

```
static void Main(string[] args)
{
int a=5, b=10;
a = a-1;
b = b+1;
a = a+5;
b = b-3;
}
```



```
static void Main(string[] args)
{
int a=5, b=10;
a--; //ou a-=1
b++; //ou b+=1
a+=5;
b-=3;
}
```

Exemplo

```
 7            int i;
 8            i=0;
 9            Console.WriteLine("i = " + i);
10            i = i+1;
11            Console.WriteLine("i = " + i);
12            i++; // igual a i = i+1;, mas só funciona de 1 em 1
13            Console.WriteLine("i = " + i);
14            i += 1; // igual a i++ e igual a i = i+1
15            Console.WriteLine("i = " + i);
16            //quero adicionar de 2 em 2 para mais
17            i += 2;
18            Console.WriteLine("i = " + i);
19            i = i + 3;
20            Console.WriteLine("i = " + i);
```

Manipulando Strings

```
37          Console.Write("Aqui, eu escrevo mas ao final ele nao quebra a linha... \n\n");
38          Console.WriteLine("Já aqui, eu escrevo e ao final eu quebro a linha");
39          //este writeline
40          Console.WriteLine();
41          //funciona exatamente como este abaixo
42          Console.Write("\n");
```

Manipulando Strings

```
7   Console.WriteLine("Primeiro projeto C#");
8   Console.WriteLine("Vamos iniciar, manipulando strings");
9   string frase = "Frase da declaração de variavel com string";
10  String frase2 = "Frase da declaração de variavel com String";
11  //Concatenando string
12  Console.WriteLine("Concatenando strings");
13  Console.WriteLine("Frase 1 = "+frase);
14  Console.WriteLine("Frase 2 = "+frase2);
15  //interpolação de string
16  Console.WriteLine("interpolando strings");
17  Console.WriteLine($"frase 1 = {frase}");
18  Console.WriteLine($"frase 2 = {frase2}");
19  //Para comentar bloco, seleciona o bloco e aperta CTRL K e após CTRL C
20  //Para descomentar bloco, seleciona bloco e aperta CTRC K e após CTRL U
```

Manipulando Strings

```
22          //converter toda a string para letras maiusculas ou minusculas
23          string maiuscula = frase.ToUpper();
24          Console.WriteLine("Frase 1 maiuscula = "+maiuscula);
25
26          string minuscula = maiuscula.ToLower();
27          Console.WriteLine("Frase 1 minuscula = "+minuscula);
28
29          //função replace que substitui algo dentro da string
30          string fraseNova = frase.Replace("Frase", "string");
31          Console.WriteLine("Frase nova = "+fraseNova);
32
33          //concatenar strings
34          string novaFrase;
35          novaFrase = "Estou unindo uma frase a outra\n" + fraseNova;
36          Console.WriteLine(novaFrase);
```

Manipulando Strings

```
38          //Verifica se comeca com uma palavra especica
39          Console.WriteLine("Verifica se inicia com frase");
40          bool verificaComeco = frase.StartsWith("frase");
41          Console.WriteLine(verificaComeco);
42          Console.WriteLine("Verifica se inicia com Frase");
43          verificaComeco = frase.StartsWith("Frase");
44          Console.WriteLine(verificaComeco);
45
46          //Verifica se termina com uma palavra especifica
47          Console.WriteLine("Verifica se termina com string");
48          bool verificaFinal = frase.EndsWith("string");
49          Console.WriteLine(verificaFinal);
```

Manipulando Strings

```
52          Console.WriteLine("Verificando se as string são iguais: ");
53          //Só para lembrar como foi nossa declaraão
54          //string frase = "Frase da declaração de variavel com string";
55          //String frase2 = "Frase da declaração de variavel com String";
56          bool testeFraseIgual = frase.Equals(frase2);
57          Console.WriteLine(testeFraseIgual);
58          Console.WriteLine("Verificando se Palavra é igual a Palavra: ");
59          string palavra = "Palavra";
60          String palavra2 = "Palavra";
61          testeFraseIgual = palavra.Equals(palavra2);
62          Console.WriteLine(testeFraseIgual);
```

Comando de entrada

- **Console.ReadLine()**

  - Na linguagem C#, a leitura irá diferir para cada tipo de dado.

Comando de entrada

- Exemplo 1: leitura de uma variável:

```
static void Main(string[] args)
{
    int x;
    x = int.Parse(Console.ReadLine());
}
```

Comando de entrada

- Exemplo 2: leitura de uma variável:

```
static void Main(string[] args)
{
    int x;
    Console.WriteLine("Digite um numero: ");
    x = int.Parse(Console.ReadLine());
}
```

- Neste exemplo, antes de ler o valor para x, mostramos um texto informativo, pedindo para que o usuário digite um valor.

- Torna nossa interface mais amigável ☺

Comando de entrada

- Exemplo 2: leitura de uma variável:

```
static void Main(string[] args)
{
    int x;
    Console.WriteLine("Digite um numero: ");
    x = int.Parse(Console.ReadLine());
    Console.WriteLine("Você digitou o numero "+x);
}
```

- E também podemos mostrar o que foi digitado anteriormente...

Comando de entrada

- E para os outros tipos de dados ?
  - Para char -> char.Parse(Console.ReadLine());
  - Para double -> double.Parse(Console.ReadLine());

Comando de entrada

- Exemplo 3: leitura de variáveis:

```
static void Main(string[] args)
{
    int x;
    char ch;
    double n2;

    Console.WriteLine("Digite um numero inteiro: ");
    x = int.Parse(Console.ReadLine());

    Console.WriteLine("Digite uma letra: ");
    ch = char.Parse(Console.ReadLine());

    Console.WriteLine("Digite um numero, pode ser com decimal: ");
    n2 = double.Parse(Console.ReadLine());

    Console.WriteLine("Você digitou o numero "+x);
    Console.WriteLine("Você digitou a letra " + ch);
    Console.WriteLine("Você o numero " + n2);
}
```

Comando de entrada

```
static void Main(string[] args)
{
    int x;
    char ch;
    double n2;

    Console.WriteLine("Digite um numero inteiro: ");
    x = int.Parse(Console.ReadLine());

    Console.WriteLine("Digite uma letra: ");
    ch = char.Parse(Console.ReadLine());

    Console.WriteLine("Digite um numero, pode ser com decimal: ");
    n2 = double.Parse(Console.ReadLine(), CultureInfo.InvariantCulture);

    Console.WriteLine("Você digitou o numero "+x);
    Console.WriteLine("Você digitou a letra " + ch);
    Console.WriteLine("Você o numero " + n2);
}
```

Atividade

- Faça a leitura de dois números double e faça as 4 operações matemáticas
- Apresente na tela os resultados