1. Utilizando apenas operações primitivas, escreva o pseudocódigo (PORTUGOL) de um programa que recebe uma cadeia de caracteres pelo teclado e exibe na tela a cadeia invertida. Por exemplo, se a cadeia informada é “ABCD”, será mostrada na tela a cadeia “DCBA” (a sua solução deve servir para qualquer cadeia de caracteres não apenas para o exemplo)

Algoritmo “InverteTexto”

Var

Rep, Texto: Vetor[] de caractere

cont, i: inteiro

Início

Escreva (“A tecla 0 termina a palavra”)

Escreva (“Digite uma palavra: ”)

Enquanto Texto[i] =! “\0” faça

Leia texto[i]

i++

cont++

FimEnquanto

Para (i=0;i<cont;i++)

Rep[i] = Texto[Cont - i]

Escreva Rep[i]

FimPara

Fim

2. Construa a sua versão em linguagem C para as funções de PRIMEIRO(S), RESTANTE(S) e CONCATENAÇÃO(S1, S2)

Primeiro

char primeiroChar (vetor[])

{

return vetor[0];

}

Restante

char\* restanteString (vetor[])

{

char String[]=(vetor[] - vetor[0]);

return String[];

}

Concatenação

char\* concatString(S1[], S2[])

{

int contS1, contS2, i;

char S3[];

while (S1[i] =! “\0”)

{

contS1++;

i++;

}

for (i=0; i<contS1;i++)

{

S3[i] = S1[i];

}

i=0;

while (S2[i] =! “\0”)

{

contS2++;

i++;

}

for (i=0; i<contS2;i++)

{

S3[i+contS1] = S2[i];

}

return S3[];

}

3. Faça uma breve pesquisa e descreva como duas linguagens de programação implementa as funções primitivas e derivadas para a manipulação de cadeias de caracteres. Aponte no seu relato as fontes que você consultou para obter essas informações.

**Python**

Você pode manipular dados textuais no Python usando o objeto str. As cadeias de caracteres são sequências imutáveis de *unicode*. O Unicode é um sistema projetado para representar todos os caracteres dos idiomas. No unicode, cada letra ou caractere é representado como um número de 4 bytes. Cada número representa um caractere exclusivo.

Para representar uma string, você a coloca entre aspas. Pode haver várias maneiras de fazer isso:

* Aspas simples, como neste exemplo: As aspas simples permitem que você incorpore aspas "duplas" em sua string.
* Aspas duplas. Por exemplo: "As aspas duplas permitem que você incorpore aspas 'simples' em sua string."
* Aspas triplas, como neste exemplo: """Triple quotes using double quotes"""", '''Triple quotes using single quotes.'''

A string entre aspas triplas permite que você trabalhe com strings de várias linhas, e todos os espaços em branco associados serão incluídos na string.

## **Fatiamento de strings em Python**

Como as cadeias de caracteres são uma sequência de caracteres, você pode acessá-las por meio de *fatiamento* e *indexação* da mesma forma que faria com as listas ou tuplas do Python. As cadeias de caracteres são indexadas em relação a cada caractere da cadeia e a indexação começa em 0:

Na cadeia de caracteres acima, o primeiro índice é C e está indexado em 0. O último caractere é um ponto final ., que é o 16º caractere da cadeia. Também é possível acessar os caracteres na direção oposta a partir de -1, o que significa que você também pode usar -1 como valor de índice para acessar . na cadeia de caracteres. Há também um espaço em branco entre Chocolate e cookie, que também faz parte da cadeia de caracteres e tem seu próprio índice, 9 nesse caso. Você pode verificar isso usando o fatiamento.

Como cada caractere em uma string Python tem um número de índice correspondente, você pode acessar e manipular strings da mesma forma que outros tipos de dados sequenciais. O fatiamento é uma técnica em Python que permite que você use um elemento específico ou um subconjunto de elementos de um objeto contêiner usando seus valores de índice. O fatiamento evita que você tenha que escrever instruções de loop para percorrer os índices da cadeia de caracteres para localizar ou acessar determinadas subcadeias.

## **Operações comuns com strings**

O fatiamento e o fatiamento de intervalo são as operações comuns que você precisaria executar em strings. Há também a concatenação de strings, que é tão simples quanto a adição.

O Python fornece muitos métodos internos ou funções auxiliares para manipular strings. Substituir uma substring, colocar determinadas palavras em maiúsculas em um parágrafo, encontrar a posição de uma string dentro de outra string são algumas das operações que você pode fazer com esses métodos internos.

Dê uma olhada em alguns deles em detalhes:

* str.capitalize()Retorna uma cópia da cadeia de caracteres com o primeiro caractere em maiúscula.
* str.islower()Retorna true se todos os caracteres da cadeia de caracteres forem minúsculos e false caso contrário.
* str.find(substring)Retorna o índice mais baixo da cadeia de caracteres em que a substring é encontrada. Você também pode especificar o índice inicial e final dentro da string onde deseja que a substring seja pesquisada. Retorna -1 se a substring não for encontrada.
* str.count(substring)Conta quantas vezes uma substring ocorre na string. Você também pode especificar o índice de início e de parada da string.
* str.isspace()Retorna True se houver apenas caracteres de espaço em branco na cadeia de caracteres e, caso contrário, falso. Caracteres de espaço em branco são os caracteres como espaço, tabulação, próxima linha, etc.

**Java**

Para representar uma cadeia de caractere, a linguagem Java não possui um tipo primitivo. Java usa para essa tarefa a classe String. Em uma visão mais técnica, String é uma classe presente no Java desde a versão 1.0 do JDK que implementa a interface CharSequence. Assim, podemos dizer que Strings nada mais são do que uma cadeia ordenada de caracteres. Para se trabalhar com texto, a classe String oferece diversos métodos utilitários. Entre esses métodos, podemos destacar alguns para exemplificar sua utilidade. O método lengh() retorna um int indicando o tamanho da String, ou seja, a quantidade de caracteres que há na String atual. É comum ser utilizado em loops for. Temos também o método charAt(índice), que Retorna o caractere que estiver na posição do índice (iniciado em zero) passado como parâmetro (public char charAt(int index)).

Uma boa parte dos programas de computador precisam representar e manipular números e também seqüências de caracteres. As necessidades numéricas são, grosso modo, atendidas pelos tipos primitivos tais como long, int e double. As linguagens de alto-nível têm diferentes abordagens para lidar com a necessidade de representar e manipular as seqüências de caracteres. Na linguagem Java as seqüências de caracteres podem ser representadas utilizando arranjos de caracteres (char arrays) ou então as cadeias de caracteres (strings). Aqui iremos dar ênfase no suporte de processamento de texto utilizando cadeias de caracteres.

Quando discutimos arranjos vimos que eles são objetos especiais. Os arranjos possuem "inicializadores" e arranjos podem ser indexados. As cadeias de caracteres são objetos especiais também, mas em um sentido um pouco diferente. Uma cadeia de caractere é o único objeto que possui uma representação no código fonte do programa, ou seja uma cadeia de caractere é o único objeto que tem um literal na linguagem Java.

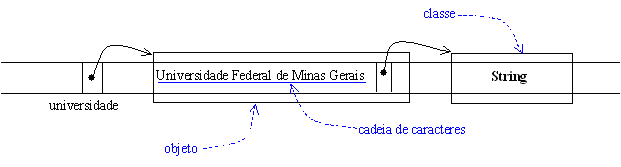
Considere o seguinte trecho de programa:  
System.out.println(123);  
System.out.println(3.1415);  
System.out.println("Ola!");

O tipo do literal 123 é int. o tipo do literal 3.1415 é double. Em Java int e double são tipos primitivos. O tipo da cadeia de caracteres "Ola!" é String. Em Java String é uma classe(class). Uma classe é o elemento da Linguagem Java que define o estado e o comportamento de seus objetos. Mais a frente no curso iremos ver as classes com mais detalhes. Neste ponto do curso iremos apenas estudar uma classe especial denominada String cujos objetos contêm cadeias de caracteres. Uma variável do tipo String é uma variável do tipo referência, ou seja, contém um endereço de objeto. Uma variável de tipo primitivo P deverá conter um valor do tipo P, uma variável do tipo referência deverá conter um endereço de um objeto do tipo referenciado, em particular uma variável do tipo String deverá conter um endereço de um objeto do tipo String.

O trecho de programa:  
String s;  
corresponde a definir uma variável com nome s e tipo String, mas o conteúdo de s não é definido. O trecho de programa:  
String s="abcde";  
 Corresponde a definir uma variável com nome s, tipo String e conteúdo correepondendo a um endereço de memória onde o compilador montou um objeto contendo a cadeia de caracteres "abcde".

Uma variável do tipo String pode ser iniciada ou inicializada de forma semelhante a uma variável de tipo primitivo. A iniciação é feita com literais do tipo String . Um literal do tipo String é uma seqüência de caracteres iniciada e terminada com aspas. Podemos utilizar o operador de concatenação de cadeias (+). O operador de concatenação permite representar uma cadeia em diferentes linhas do texto do programa. Exemplo de declaração de variáveis do tipo cadeia de caracteres já inicializadas:  
String s1="Universidade ";  
String s2="Federal de";  
String s3=" Minas "+  
 "Gerais";  
String universidade=s1+s2+s3;

No trecho de programa acima a variável universidade é inicializada utilizando o operador de concatenação de cadeias de caracteres. Os literais de tipo String podem ser usados em comandos de atribuição:  
String universidade;  
universidade="Universidade Federal de Minas Gerais";

No trecho acima a variável universidade recebe o endereço de memória onde fica um objeto contendo a cadeia de caracteres "Universidade Federal de Minas Gerais". A figura abaixo mostra uma representação gráfica simplificada para o que discutimos acima:  


**Bibliografia**

18/02/2025 - 22:00 as 22:22

[**https://www.datacamp.com/pt/tutorial/python-string-tutorial**](https://www.datacamp.com/pt/tutorial/python-string-tutorial)

[**https://brainly.com.br/tarefa/60448293**](https://brainly.com.br/tarefa/60448293)

[**https://homepages.dcc.ufmg.br/~rodolfo/aeds-1-05/cadeiadecaractere/cadeiadecaractere.html**](https://homepages.dcc.ufmg.br/~rodolfo/aeds-1-05/cadeiadecaractere/cadeiadecaractere.html)