Lista 1 — Respostas

Linguagens Formais de Autômatos

Henrique de Souza Lima

2024/2

Problema 1

Enunciado

Escreva um programa que converte um número inteiro de uma base numérica para outra. O programa deve receber como entrada o número a ser convertido, a base original e a base para a qual ele deve ser convertido. As bases suportadas devem ser 2 (binária), 8 (octal), 10 (decimal), 12 (duodecimal), 16 (hexadecimal), e 20 (vigesimal).1 (rodapé: 1Para base numéricas maiores do que 10, use as letras maiúsculas no alfabeto latino como os dígitos adicionais. Por exemplo, os dígito da base 16 serão 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.)

Solução

A solução adotada foi implementar 2 funções, sendo uma auxiliar com uso de recursão. Primeiramente, converte-se a string recebida por um número inteiro, multiplicando cada dígito pelo seu valor correspondente, associado através de um Map(Char, Int), e da posição que ocupa, invertendo a string recebida para que os indices estejam corretamente relacionados com o expoente daquela base; por fim, soma-se todos os valores obtidos. Com o resultado, chama-se a função recursiva de cauda com o numero e sua base final, cujo caso base é quando o número em seu parametro é igual a 0. Nos demais casos, o número é dividido sucessivamente pela base desejada e possui o caracter associado ao o resto desta divisão acrescentado ao início de uma string, iniciada vazia. Ao fim, obtém-se a convertido para a base desejada.

Problema 2

Enunciado

Desenvolva um programa que verifica se uma expressão composta apenas de parênteses (‘(’ e ‘)’) está corretamente balanceada, ou seja, se cada parêntese de abertura tem um correspondente parêntese de fechamento. O programa receberá como entrada uma string

contendo apenas parênteses, e como saída uma mensagem que indica se a expressão está bem formada.

Solução

A solução adotada foi implementar uma recursão. O caso base é quando a string é completamente percorrida ou quando o número de parêntesis abertos torna-se negativo. Caso tenha sido completamente percorrida, verifica se o contador de parentêsis abertos (rest, passado como parametro) é igual a 0, indicando que cada parentêsis aberto (+1 ao contador) possui correspondente fechado (-1 ao contador), o qual mostra tratar-se de uma expressão bem formada. Caso contrário, envia mensagens adequadas à razão da falta de correspondência. O contador que percorre os caracteres da string inicia-se com 0 e é passado como parametro, sendo incrementado a cada iteração.

Problema 3

Enunciado

Escreva um programa que verifique se um número é perfeito. Um número perfeito é aquele que é igual à soma de seus divisores próprios, i.e., os divisores excluindo o um o próprio número. O programa receberá como entrada um número inteiro positivo, e como saída uma mensagem que indica se o número é perfeito ou não.

Solução

A solução adotada foi implementar uma recursão. O caso base é quando o divisor atual, inicialmente igual a 1, é igual ou maior que o número passado inicialmente para a função. Caso seja, verifica se a soma dos divisores, positivos, do número desejado, exceto ele mesmo, é igual ao próprio número, e indica, em uma mensagem, se é ou não perfeito em conformidade com esta condição. O conjunto dos divisores, implementado como set, inicia-se vazio. Nos demais casos, verifica-se se o resto da razão inteira entre do número e o divisor atual é igual ou não a zero, e acrescentará este divisor ao set em caso afirmativo; em seguida, soma o valor do divisor atual por 1. O resultado é um set contendo todos os divisores desse número.

Problema 4

Enunciado

Crie um programa que calcula o valor de uma expressão da álgebra de conjuntos. O programa deve permitir ao usuário inserir conjuntos (apenas conjuntos definidos por extensão, i.e., listando todos os elementos) e, em seguida, o programa deve receber como entrada uma string que representa a expressão e retornar o resultado. A expressão em questão poderá usar:

- União ($A \cup B$);
- Interseção ($A \cap B$);
- Diferença ($A - B$);
- Diferença simétrica ($A \Delta B$);
- Complemento ($\sim A$);
- Produto cartesiano ($A \times B$);
- Conjunto das partes ($P(A)$).

Exemplos simples:

- $A \cup (B \cap C)$: União do conjunto A com a interseção dos conjuntos B e C.
- $(A - B) \Delta C$: Diferença simétrica entre a diferença de A e B e o conjunto C.
- $\sim (A \cup B)$: Complemento da união dos conjuntos A e B.
- $P(A) - P(B)$: Diferença entre o conjunto das partes de A e o conjunto das partes de B.
- $A \times (B \cup C)$: Produto cartesiano de A com a união de B e C.

Exemplos mais complexos:

- $\sim ((A \cap B) \cup (C - D))$: Complemento da união entre a interseção de A e B e a diferença de C e D.
- $P(A \cap B) \Delta P(A \cup B)$: Diferença simétrica entre o conjunto das partes da interseção de A e B e o conjunto das partes da união de A e B.
- $(A \times B) \cap (A \times C)$: Interseção do produto cartesiano de A e B com o produto cartesiano de A e C.
- $\sim (\sim A \cap \sim B)$: Complemento da interseção dos complementos de A e B (equivalente a $A \cup B$ pela Lei de De Morgan).

(O conjunto universo foi definido como a união de todos os conjuntos definidos.)

Solução

A solução adotada foi adotar uma classe SetAlgebra que contém um traço (sealed trait) chamado SetResult, que é estendido por StringSet e PowerSet, para permitir resultados em forma de Sets de strings ou Sets de Sets de Strings, como P(A). O programa possui funções para cada operação de conjuntos, sendo chamadas de acordo com a ocorrência de caracteres definidos, (Diferença, Conjunto das Partes, etc.), cuja assinatura, em inglês, corresponde ao seu produto. A função `evaluate_expression` analisa uma expressão em

notação de infixos a partir recebida como uma String, que é convertida em uma lista de tokens e avaliada segundo a precedência devida. Para cada token, se for um conjunto definido pelo usuário, ele é empilhado; se for um operador, as operações correspondentes são aplicadas aos conjuntos no topo da pilha. Para expressões mais complexas, fez-se utilização de uma pilha de operadores de modo que as operações ocorram na ordem devida para fazer a aplicação das operações das expressões entre parênteses. Os parâmetros são pedidos no o programa principal. Primeiro, a quantidade de conjuntos (até 4, nomeados de A a D); em seguida, lerá uma linha dos elementos do conjunto respectivo – obtidos através da separação por espaço em branco entre caracteres (um split), seguido de toSet.