# PL1-Racket

## Início

|  |  |
| --- | --- |
|  | **Disciplina**: AARE Paradigmas de Linguagens de Programação  **Professor**: Prof. Ausberto S. Castro V.  **E-mail:** ascv@uenf.br  **Data**: 9 de setembro de 2021 |

**Nome Completo:** João Vítor Fernandes Dias

**Data:** 9 de setembro de 2021 **Total Exercícios Resolvidos**: Todos

### -------------------------✂ cortar aqui depois de ler ✂---------------------------------------------

**Instruções:**

* O **relatório** desta prática, deverá conter claramente o **Nome Completo** do aluno e a última **data** do trabalho no laboratório
* Cada item resolvido deverá conter um gráfico da **imagem capturada** da janela de execução ou janela gráfica ou do editor, onde deve aparecer obrigatoriamente o nome do aluno, e as mudanças realizadas.
* **Em todas as questões desta prática**, apresentar o código fonte modificado ou atualizado e a execução do programa (imagem capturada). Crie um **NOVO** arquivo, para cada mudança de código e mantenha o arquivo original. ArquivoN modificado deve ser renomeado para ArquivoNa, ArquivoNb, ArquivoNc, ...
* Em todos os arquivos use a linguagem R5RS, preferencialmente, exceto quando implicitamente indicado outra linguagem
* Incluir, neste relatório, os códigos fonte modificados

**Software: DrRacket (**Instale o software em idioma Inglês)

* [http://racket-lang.org/](%20http://racket-lang.org/)
* <http://schemers.org/> (referências)

**Livro On-Line:** <http://www.htdp.org/> (The Book)

**Captura de gráficos**

* <Alt>+<Print Screen> ou
* Gadwin PrintScreen (<http://www.gadwin.com/printscreen/>)

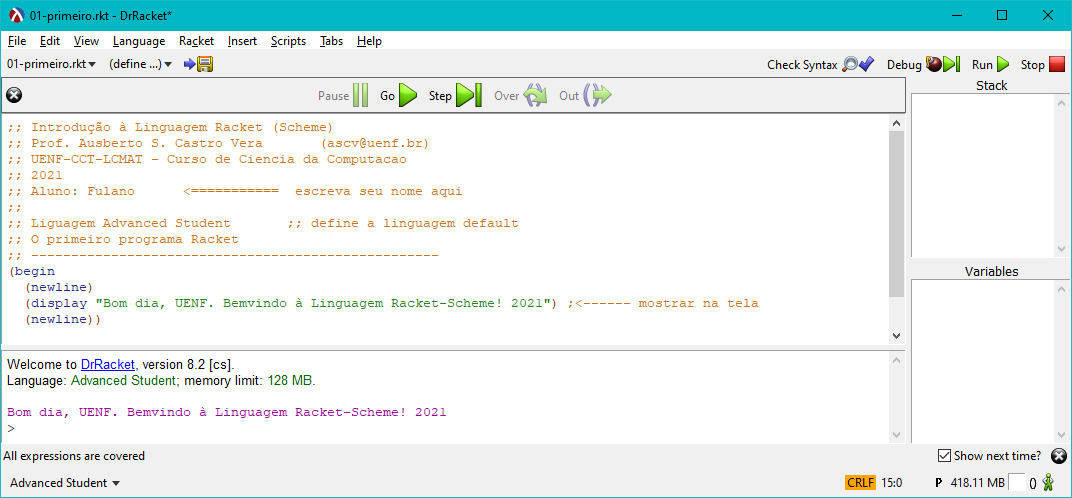
-------------------------✂ cortar aqui depois de ler ✂------------------------------------

## Exercícios

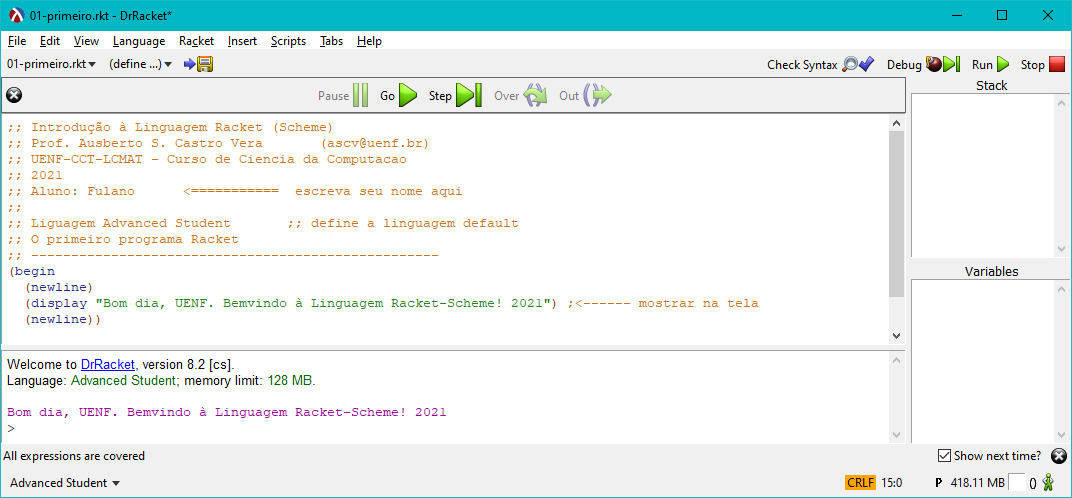
#### Arquivo 01-primeiro.rtk Primeiro programa em Racket

##### Execute o programa e indicar o que faz cada linha do código fonte do programa. Quais funções estão definidas nas 4 linhas. Explique cada uma delas.

###### Imagens Ausberto



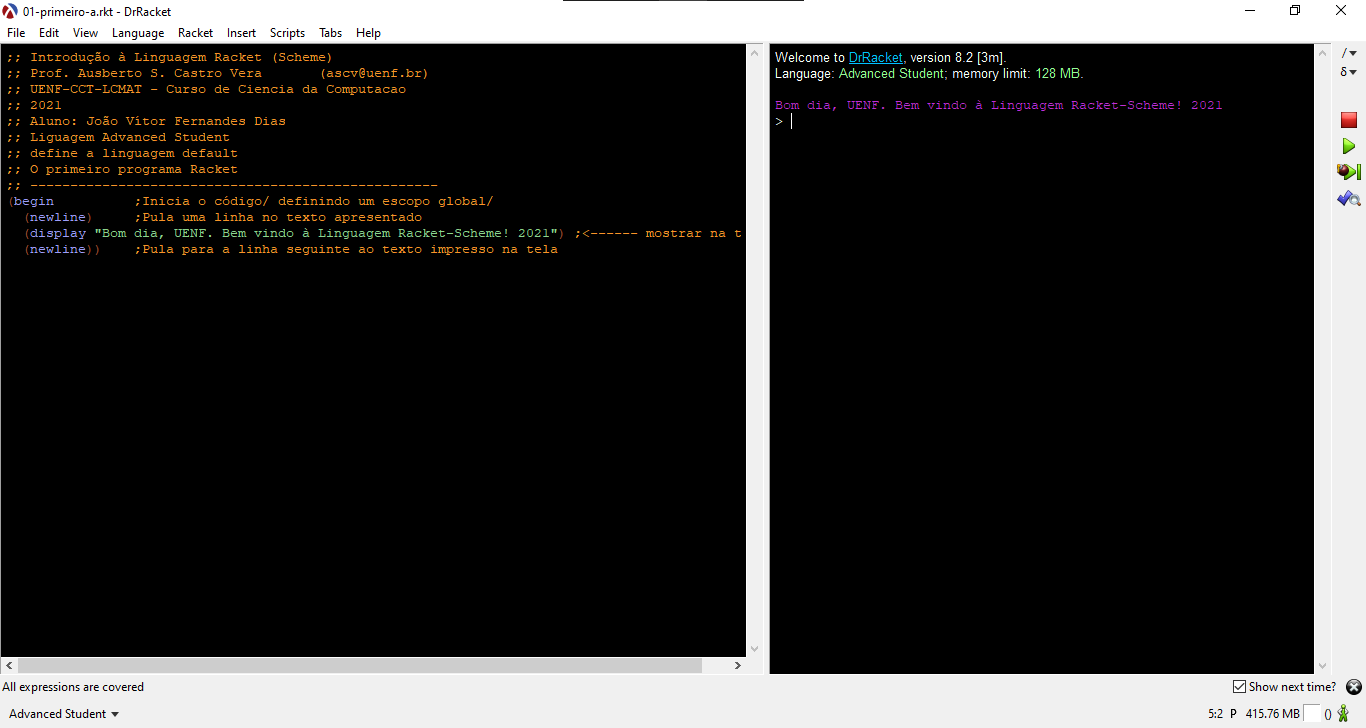
Codigo fonte



Interações (shell)

###### Explicar aqui:

Print



Código:

(begin ;Inicia o código/ definindo um escopo global/

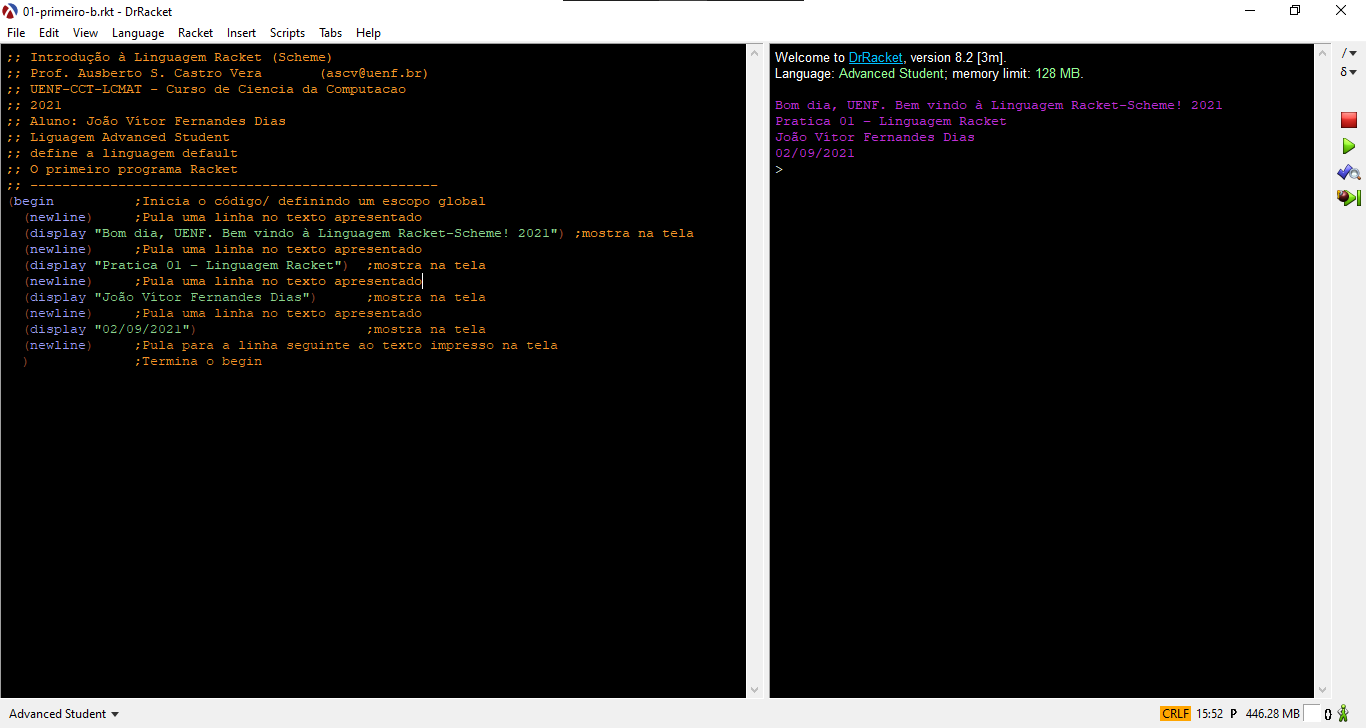
(newline) ;Pula uma linha no texto apresentado

(display "Bom dia, UENF. Bem vindo à Linguagem Racket-Scheme! 2021") ;<------ mostrar na tela

(newline)) ;Pula para a linha seguinte ao texto impresso na tela

##### Agregar linhas de código para mostrar na parte executável, a mensagem “Pratica 01 – Linguagem Racket”, o nome completo do aluno e a data atual

###### Print



###### Código

(begin ;Inicia o código/ definindo um escopo global

(newline) ;Pula uma linha no texto apresentado

(display "Bom dia, UENF. Bem vindo à Linguagem Racket-Scheme! 2021") ;mostra na tela

(newline) ;Pula uma linha no texto apresentado

(display "Pratica 01 – Linguagem Racket") ;mostra na tela

(newline) ;Pula uma linha no texto apresentado

(display "João Vítor Fernandes Dias") ;mostra na tela

(newline) ;Pula uma linha no texto apresentado

(display "02/09/2021") ;mostra na tela

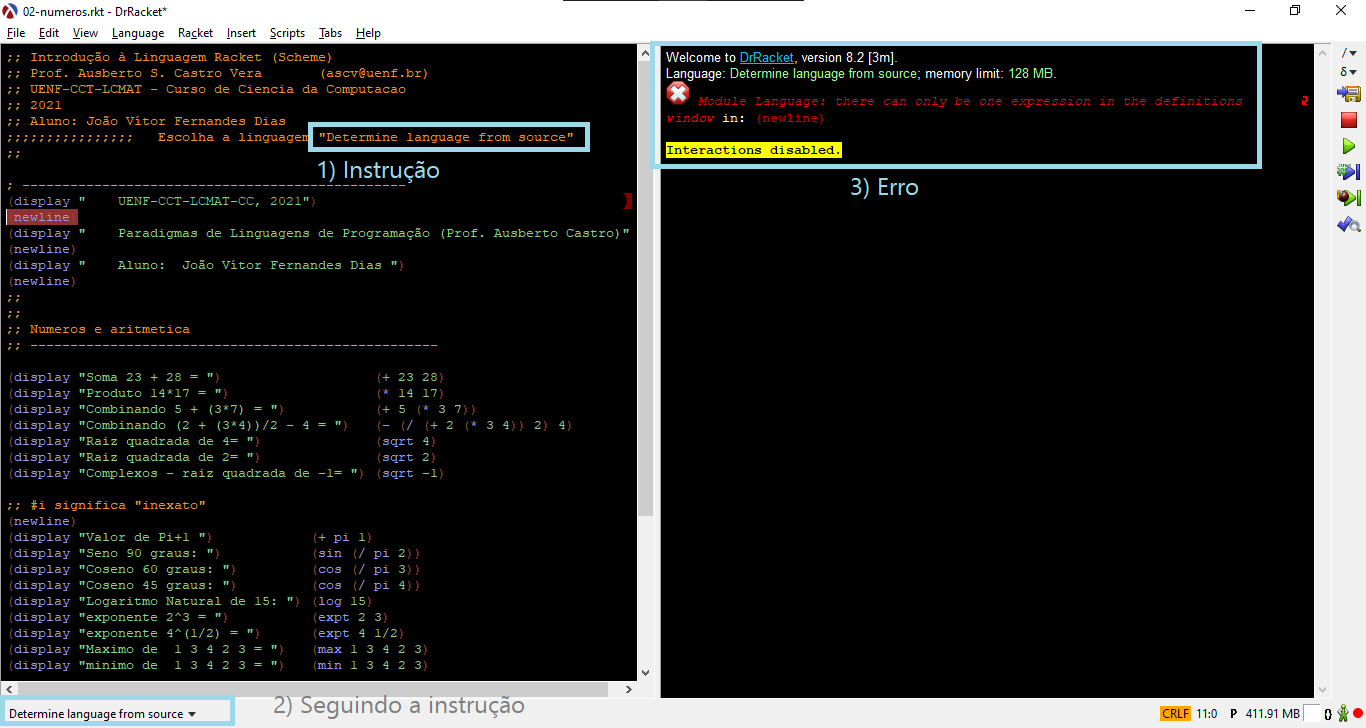
(newline) ;Pula para a linha seguinte ao texto impresso na tela

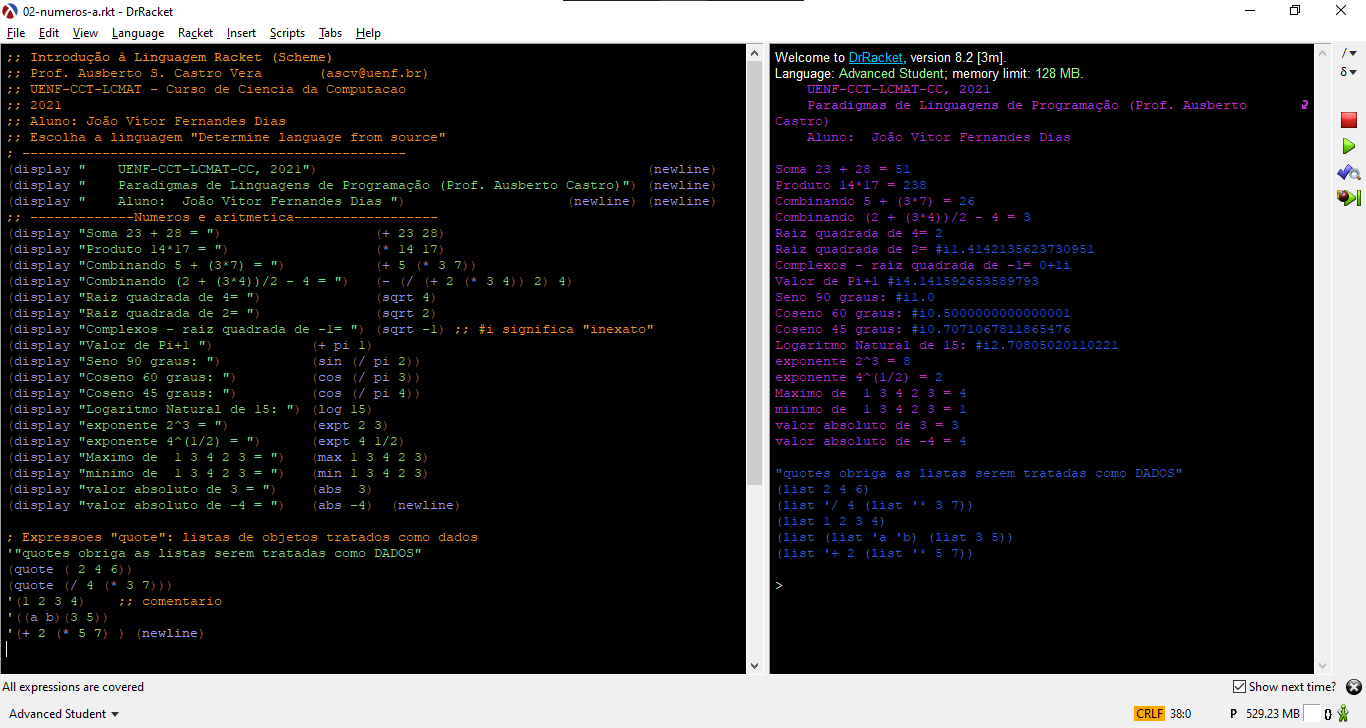
) ;Termina o begin

#### Arquivo 02-numeros.rtk Números e Aritmética DÚVIDA

##### Execute o programa e mostre os resultados

###### Print





###### Código:

;; -------------Numeros e aritmetica------------------

(display "Soma 23 + 28 = ") (+ 23 28)

(display "Produto 14\*17 = ") (\* 14 17)

(display "Combinando 5 + (3\*7) = ") (+ 5 (\* 3 7))

(display "Combinando (2 + (3\*4))/2 - 4 = ") (- (/ (+ 2 (\* 3 4)) 2) 4)

(display "Raiz quadrada de 4= ") (sqrt 4)

(display "Raiz quadrada de 2= ") (sqrt 2)

(display "Complexos - raiz quadrada de -1= ") (sqrt -1) ;; #i significa "inexato"

(display "Valor de Pi+1 ") (+ pi 1)

(display "Seno 90 graus: ") (sin (/ pi 2))

(display "Coseno 60 graus: ") (cos (/ pi 3))

(display "Coseno 45 graus: ") (cos (/ pi 4))

(display "Logaritmo Natural de 15: ") (log 15)

(display "exponente 2^3 = ") (expt 2 3)

(display "exponente 4^(1/2) = ") (expt 4 1/2)

(display "Maximo de 1 3 4 2 3 = ") (max 1 3 4 2 3)

(display "minimo de 1 3 4 2 3 = ") (min 1 3 4 2 3)

(display "valor absoluto de 3 = ") (abs 3)

(display "valor absoluto de -4 = ") (abs -4) (newline)

; Expressoes "quote": listas de objetos tratados como dados

'"quotes obriga as listas serem tratadas como DADOS"

(quote ( 2 4 6))

(quote (/ 4 (\* 3 7)))

'(1 2 3 4) ;; comentario

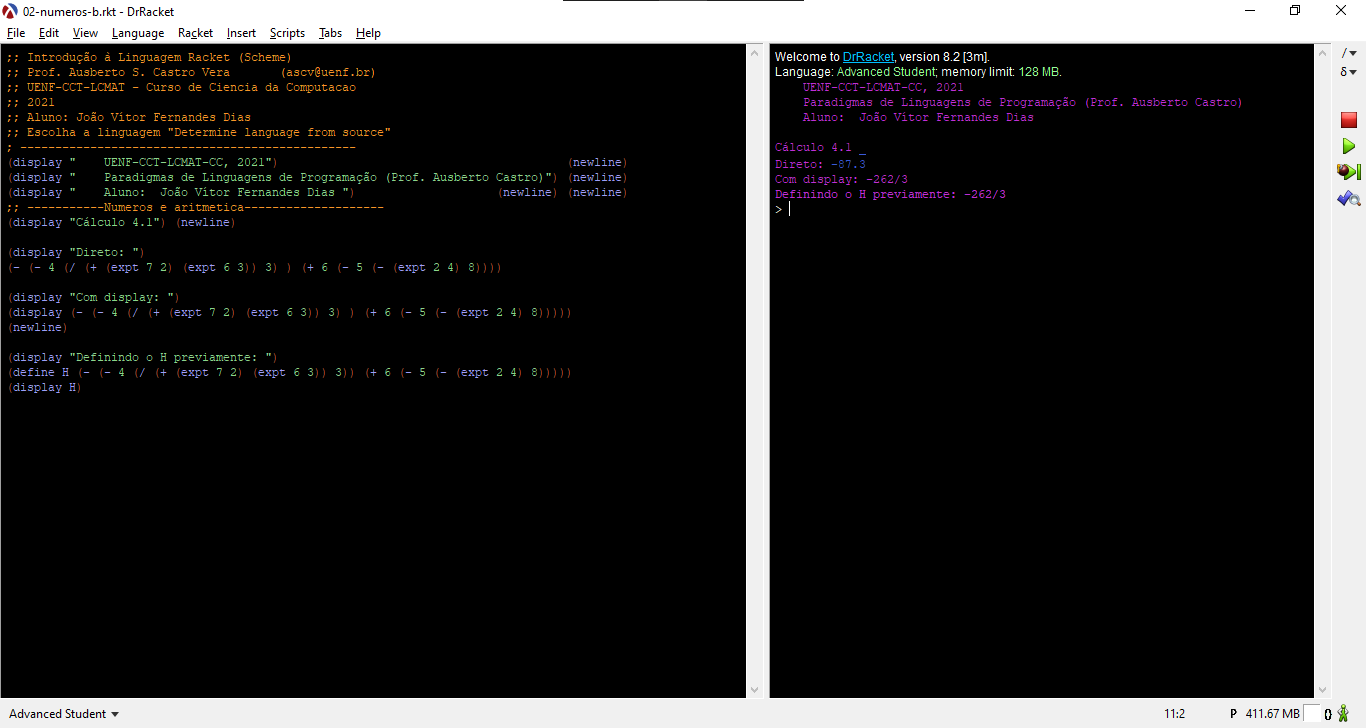
'((a b)(3 5))

'(+ 2 (\* 5 7) ) (newline)

##### Escreva programas Racket para as seguintes expressões:

###### H = (4 – (7^2 + 6^3) / 3) – (6 + (5 – (2^4 – 8))

Print



Código:

(display "Cálculo 4.1") (newline)

(display "Direto: ")

(- (- 4 (/ (+ (expt 7 2) (expt 6 3)) 3) ) (+ 6 (- 5 (- (expt 2 4) 8))))

(display "Com display: ")

(display (- (- 4 (/ (+ (expt 7 2) (expt 6 3)) 3) ) (+ 6 (- 5 (- (expt 2 4) 8)))))

(newline)

(display "Definindo o H previamente: ")

(define H (- (- 4 (/ (+ (expt 7 2) (expt 6 3)) 3)) (+ 6 (- 5 (- (expt 2 4) 8)))))

(display H)

###### Escreva um NOVO programa Racket que calcule o valor da expressão:

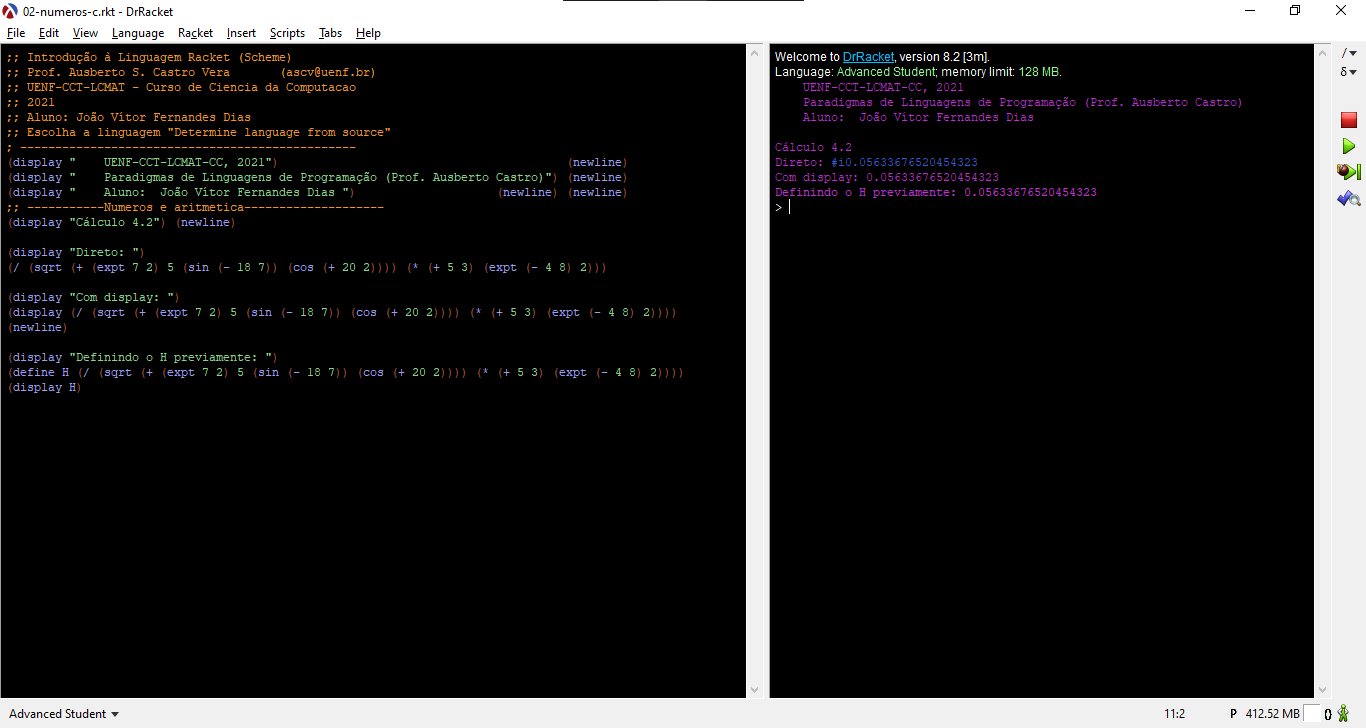
Expressão



sqrt (7^2+5+sin (18-7) +cos (20+2)) / ((5+3) \*(4-8)^2)



Print

****

Código:

(display "Cálculo 4.2") (newline)

(display "Direto: ")

(/ (sqrt (+ (expt 7 2) 5 (sin (- 18 7)) (cos (+ 20 2)))) (\* (+ 5 3) (expt (- 4 8) 2)))

(display "Com display: ")

(display (/ (sqrt (+ (expt 7 2) 5 (sin (- 18 7)) (cos (+ 20 2)))) (\* (+ 5 3) (expt (- 4 8) 2))))

(newline)

(display "Definindo o H previamente: ")

(define H (/ (sqrt (+ (expt 7 2) 5 (sin (- 18 7)) (cos (+ 20 2)))) (\* (+ 5 3) (expt (- 4 8) 2))))

(display H)

#### Arquivo 03-variaveis.rtk

##### Execute o programa e indique o valor das variáveis m, z, k

###### Print



Código:

;; Variáveis e Expressoes LET

;; ---------------------------------------------------

(define x 3) ; para x=3

(define m 7)

(define z (+ m 6)) ; para z = m+ 6

(define Pi 3.141516)

(define k (+ z (- m x)))

(display "m: ") (display m)

(display "z: ") (display z)

(display "k: ") (display k)

#|

m=7

z=m+6=7+6=13

k=z+(m-x)=13+(7-3)=13+4=17

m=7 z=13 k=17

|#

; (define (nomeFuncao parametro) definicao )

(define (quadrado x) (\* x x) )

(define (entre5e20? n) (and (< 5 n) (< n 20) )) ; AND logico

;; ---------------------------------------------------

(display "O numero 8 esta entre 5 e 20? ") (entre5e20? 8) (newline)

(display "O numero 37 esta entre 5 e 20? ") (entre5e20? 37) (newline)

(display "Escreva qualquer variavel definida acima ... <var> <ENTER> x m z Pi") (newline)(newline)

;; ---------------------------------------------------

; Expressao let:

; (let ((var1 valor) .. ) expr1 expr2 .....)

(let ((x 24 ) ) (+ x 6) )

#|

Permite que o X tenha o valor 24

dentro da expressão "+ x 6",

assim retornando o valor 24+6=30

|#

(let ((a 5) (b 8) ) (+ 3 (\* a b)) );

#|

Permite que o a tenha o valor 5 e que b tenha o valor 8

dentro da expressão "+ 3 (\* a b)",

assim retornando o valor 3+5\*8=43

|#

(let ((op1 +) (op2 \*) (x 5) ) (op1 3 (op2 4 x)) )

#|

Permite que o op1 tenha o valor "+", op2 tenha o valor "\*" e x tenha o valor 5

dentro da expressão "(op1 3 (op2 4 x)",

obtendo então a expressão "+ 3 (\* 4 5)",

assim retornando o valor 3+4\*5 = 23

|#

(display "O quadrado de 12 : ")(quadrado 12)

(let ((a 1) (b 2)) (+ a b))

#|

Define o valor de a como sendo 1 e o valor de b como sendo 2

Então faz a soma de a com b

Retornando o valor a+b=1+2=3

|#

(let ((b 6) (c 8)) (- c b))

#|

Define o valor de b como sendo 6 e o valor de c como sendo 8

Então faz a subtração de b em c

Retornando o valor c-b=8-6=2

|#

#| ;Tentativa de Bhaskara

(let ((a 1) (b 3) (c 5))

(

(let ((vezes \*) (mais +) (divide /) (subtrai -) (quatro 4) (dois 2))

(

(let ( (delta (subtrai (vezes b b) (vezes quatro a c ) )))

(

( let ((raizDelta (sqrt delta)) (menosB (- 0 b)) (doisA (vezes dois a)))

(

(let ((menosBsobredoisA (divide menosB doisA)) (raizDeltasobredoisA (divide raizDelta doisA)))

(

(let ((X1 (mais menosBsobredoisA raizDeltasobredoisA)) (X2 (subtrai menosBsobredoisA raizDeltasobredoisA)))

(

(let ((teste1 0))

(

+ X1 0

)

)

)

)

)

)

)

)

)

)

)

)

)

)

|#

#|

(let ((teste2 0))

(

+ X2 0

)

)

|#

#|

(newline)

(display "Resolvendo Bhaskara da equação: (") (display a) (display ")X²+(") (display b)

(display ")+(") (display c) (display ")") (newline) (newline)

(display "X1 = ") X1

(display "X2 = ") X2 (newline) (newline)

|#

###### Valores das variáveis m=7; z=13; k=17

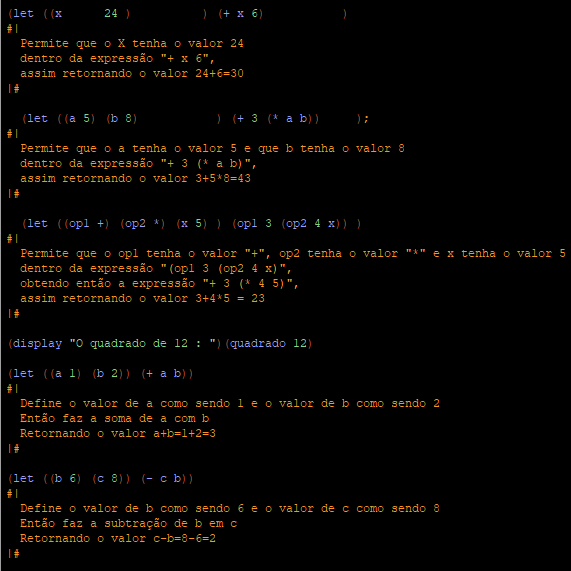
m=7

z=m+6=7+6=13

k=z+(m-x) =13+(7-3) =13+4=17

###### Explicar o significado de cada uma das 3 expressões de iteração let

Print



Explicações:

(let ((x 24 ) ) (+ x 6) )

#|

Permite que o X tenha o valor 24

dentro da expressão "+ x 6",

assim retornando o valor 24+6=30

|#

(let ((a 5) (b 8) ) (+ 3 (\* a b)) );

#|

Permite que o a tenha o valor 5 e que b tenha o valor 8

dentro da expressão "+ 3 (\* a b)",

assim retornando o valor 3+5\*8=43

|#

(let ((op1 +) (op2 \*) (x 5) ) (op1 3 (op2 4 x)) )

#|

Permite que o op1 tenha o valor "+", op2 tenha o valor "\*" e x tenha o valor 5

dentro da expressão "(op1 3 (op2 4 x)",

obtendo então a expressão "+ 3 (\* 4 5)",

assim retornando o valor 3+4\*5 = 23

|#

(display "O quadrado de 12 : ")(quadrado 12)

(let ((a 1) (b 2)) (+ a b))

#|

Define o valor de a como sendo 1 e o valor de b como sendo 2

Então faz a soma de a com b

Retornando o valor a+b=1+2=3

|#

(let ((b 6) (c 8)) (- c b))

#|

Define o valor de b como sendo 6 e o valor de c como sendo 8

Então faz a subtração de b em c

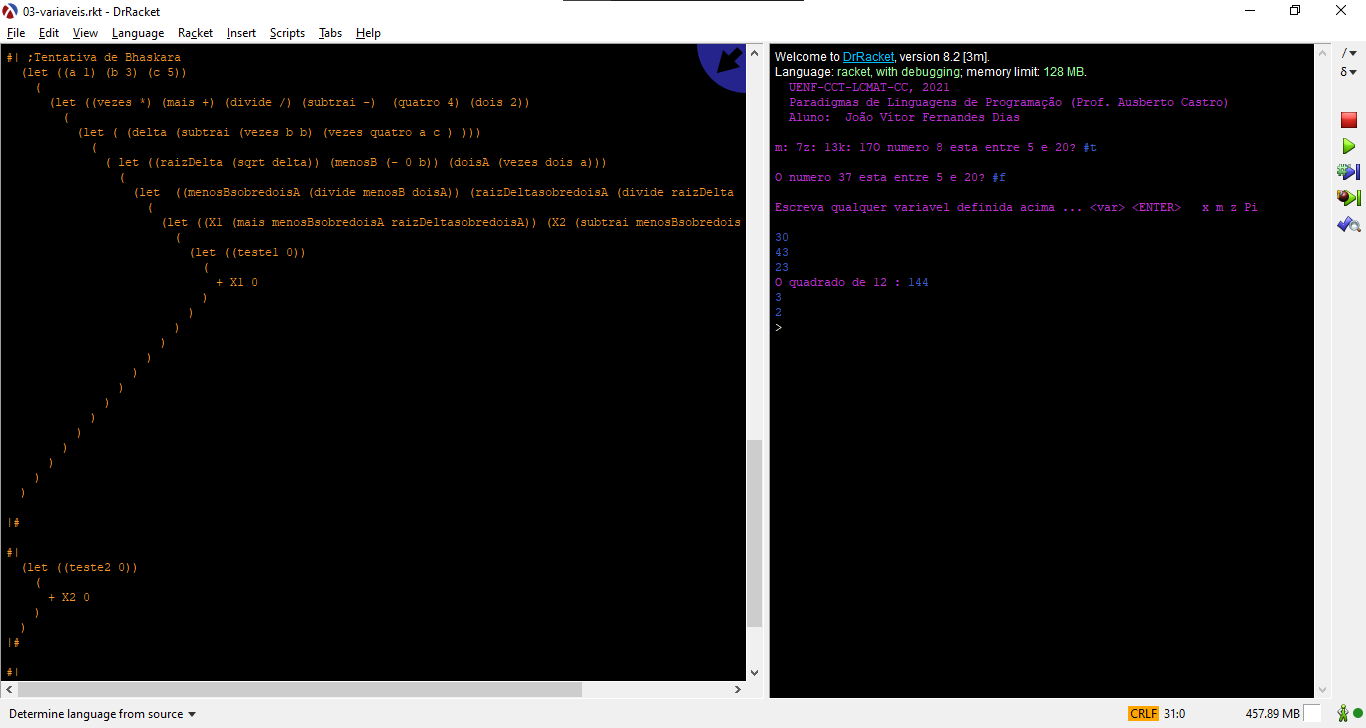
Retornando o valor c-b=8-6=2

|#

##### Escreva 2 expressões do tipo let e explique o seu significado

###### Primeira Tentativa – Bhaskara – Não consegui

Print



Código

#| ;Tentativa de Bhaskara

(let ((a 1) (b 3) (c 5))

(

(let ((vezes \*) (mais +) (divide /) (subtrai -) (quatro 4) (dois 2))

(

(let ( (delta (subtrai (vezes b b) (vezes quatro a c ) )))

(

( let ((raizDelta (sqrt delta)) (menosB (- 0 b)) (doisA (vezes dois a)))

(

(let ((menosBsobredoisA (divide menosB doisA)) (raizDeltasobredoisA (divide raizDelta doisA)))

(

(let ((X1 (mais menosBsobredoisA raizDeltasobredoisA)) (X2 (subtrai menosBsobredoisA raizDeltasobredoisA)))

(

(let ((teste1 0))

(

+ X1 0

)

)

)

)

)

)

)

)

)

)

)

)

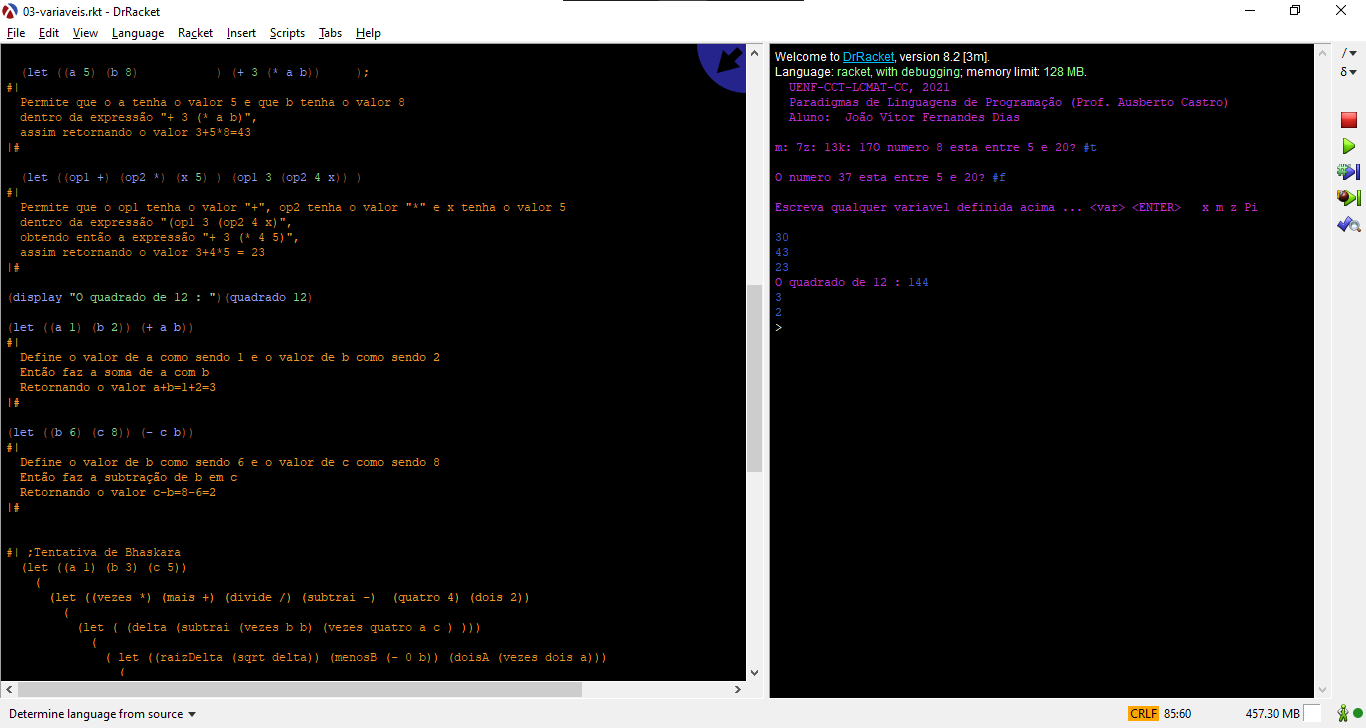
)

)

|#

###### Segunda tentativa – Mais simples

Print



Código

(let ((a 1) (b 2)) (+ a b))

#|

Define o valor de a como sendo 1 e o valor de b como sendo 2

Então faz a soma de a com b

Retornando o valor a+b=1+2=3

|#

(let ((b 6) (c 8)) (- c b))

#|

Define o valor de b como sendo 6 e o valor de c como sendo 8

Então faz a subtração de b em c

Retornando o valor c-b=8-6=2

|#

#### Arquivo 04-areas.rtk

##### Execute o programa e indique o que faz o programa

###### Print



Explicação

Esse código define fórmulas para cálculo de áreas dos planos: Círculo, triângulo, disco.

Abaixo das definições, elas são testadas com alguns valores.

###### Código

;; ---------------AREAS-----------------------------

; (define (nomeFuncao parametros) definicao )

(define (area-circulo r) (\* 3.14 (\* r r)) )

(define (areatriangulo b h) (/ (\* b h) 2) )

(define (areadisco interno externo) (- (area-circulo externo) (area-circulo interno) ) )

;--------------------------------------------------------------

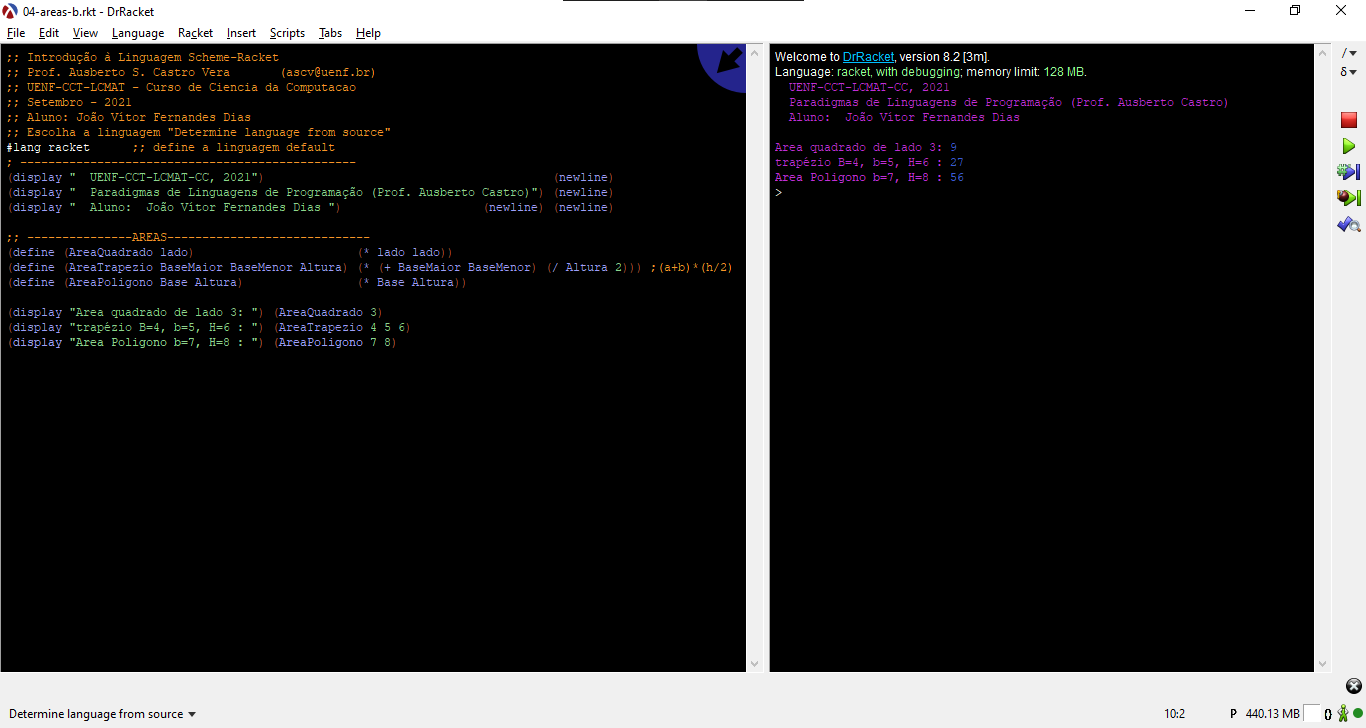
(display "Circulo de raio 18 AREA = ") (area-circulo 18)

(display "Triangulo base=10, altura=15 AREA = ") (areatriangulo 10 15)

(display "Disco raio menor=12, raio maior=20 AREA = ") (areadisco 12 20)

##### Escreva um programa Racket para calcular a área de um quadrado qualquer, a área de um trapézio e a área de um polígono.

###### Print



###### Código

(define (AreaQuadrado lado) (\* lado lado))

(define (AreaTrapezio BaseMaior BaseMenor Altura) (\* (+ BaseMaior BaseMenor) (/ Altura 2))) ;(a+b)\*(h/2)

(define (AreaPoligono Base Altura) (\* Base Altura))

(display "Area quadrado de lado 3: ") (AreaQuadrado 3)

(display "trapézio B=4, b=5, H=6 : ") (AreaTrapezio 4 5 6)

(display "Area Poligono b=7, H=8 : ") (AreaPoligono 7 8)

##### fórmula V = πR2A, onde as variáveis V, R e A representam, respectivamente, o volume, o raio e a altura

###### Print



###### Código

(define (VolumeOleo R A) (\* pi R R A ) )

(display "Volume do barril de óleo de raio 2 e altura 4: ")

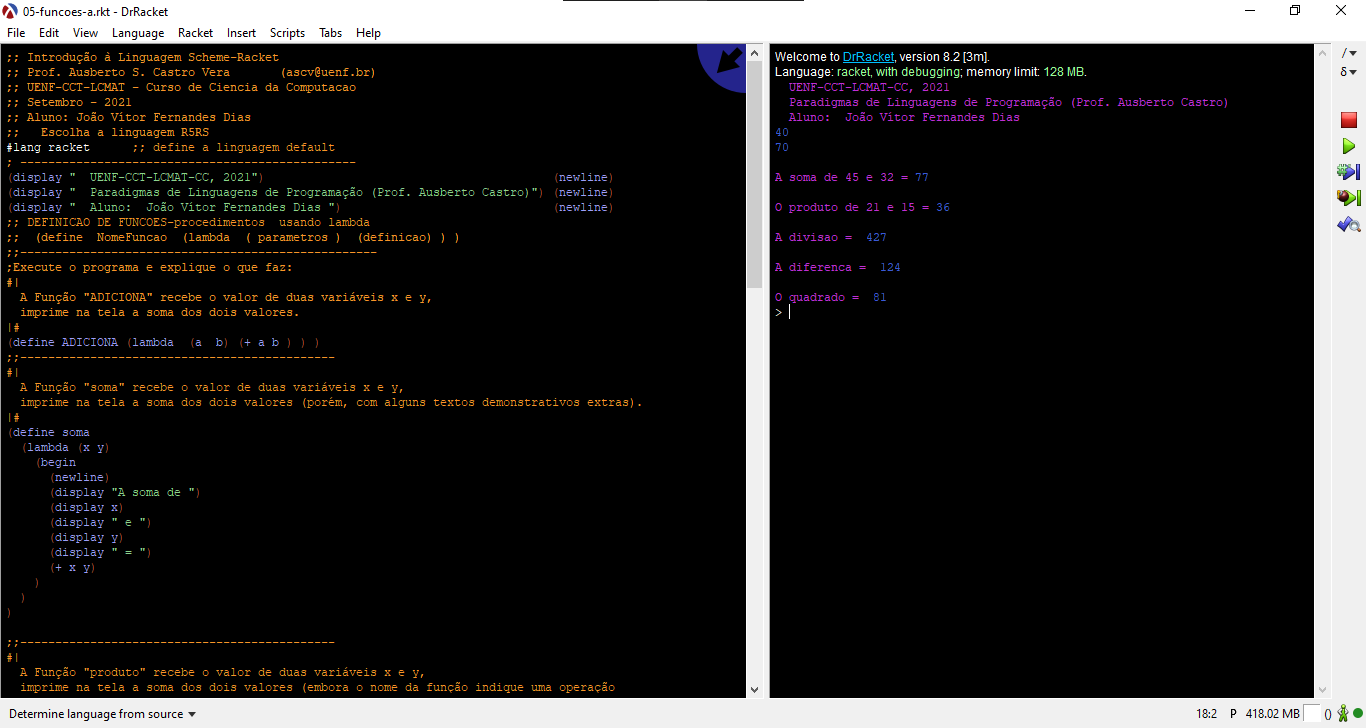
(define V (VolumeOleo 2 4))

(display V)

#### Arquivo 05-funcoes.rtk

##### Execute o programa e explique o que faz

###### Print



###### Código

#|

A Função "ADICIONA" recebe o valor de duas variáveis x e y,

imprime na tela a soma dos dois valores.

|#

(define ADICIONA (lambda (a b) (+ a b ) ) )

;;---------------------------------------------

#|

A Função "soma" recebe o valor de duas variáveis x e y,

imprime na tela a soma dos dois valores (porém, com alguns textos demonstrativos extras).

|#

(define soma

(lambda (x y)

(begin

(newline)

(display "A soma de ")

(display x)

(display " e ")

(display y)

(display " = ")

(+ x y)

)

)

)

;;---------------------------------------------

#|

A Função "produto" recebe o valor de duas variáveis x e y,

imprime na tela a soma dos dois valores (embora o nome da função indique uma operação

diferente (Não sei se eu deveria alterar a operação para que ela faça o que diz de fato))

|#

(define produto

(lambda (x y)

(begin

(newline)

(display "O produto de ") (display x) (display " e ") (display y)

(display " = ")

(+ x y)

)

)

)

;;---------------------------------------------

(define divisao

(lambda (x y)

(begin

(newline)

(display "A divisao = ")

(+ x y)

)

)

)

#|

A Função "divisao" recebe o valor de duas variáveis x e y,

imprime na tela a soma dos dois valores (embora o nome da função indique uma operação

diferente (Não sei se eu deveria alterar a operação para que ela faça o que diz de fato))

|#

;;---------------------------------------------

#|

A Função "diferenca" recebe o valor de duas variáveis x e y,

imprime na tela a soma dos dois valores (embora o nome da função indique uma operação

diferente (Não sei se eu deveria alterar a operação para que ela faça o que diz de fato))

|#

(define diferenca

(lambda (x y)

(begin

(newline)

(display "A diferenca = ")

(+ x y)

)

)

)

;;---------------------------------------------

#|

A Função "quadrado" recebe o valor de uma variável a,

imprime na tela o produto de a com a, ou seja, eleva a ao quadrado.

|#

(define quadrado

(lambda ( a )

(newline)

(display "O quadrado = ")

( \* a a)

)

)

;;---------------------------------------------

;;----------- Executando funcoes --------------

#|

Nessa seção são feitos alguns testes de valores nas funções

|#

(ADICIONA 25 15)

(ADICIONA 30 40)

(soma 45 32)

(produto 21 15)

(divisao 420 7)

(diferenca 89 35)

(quadrado 09)

###### Explicação

São definidas algumas funções:

ADICIONA, soma, produto, divisão, diferença e quadrado.

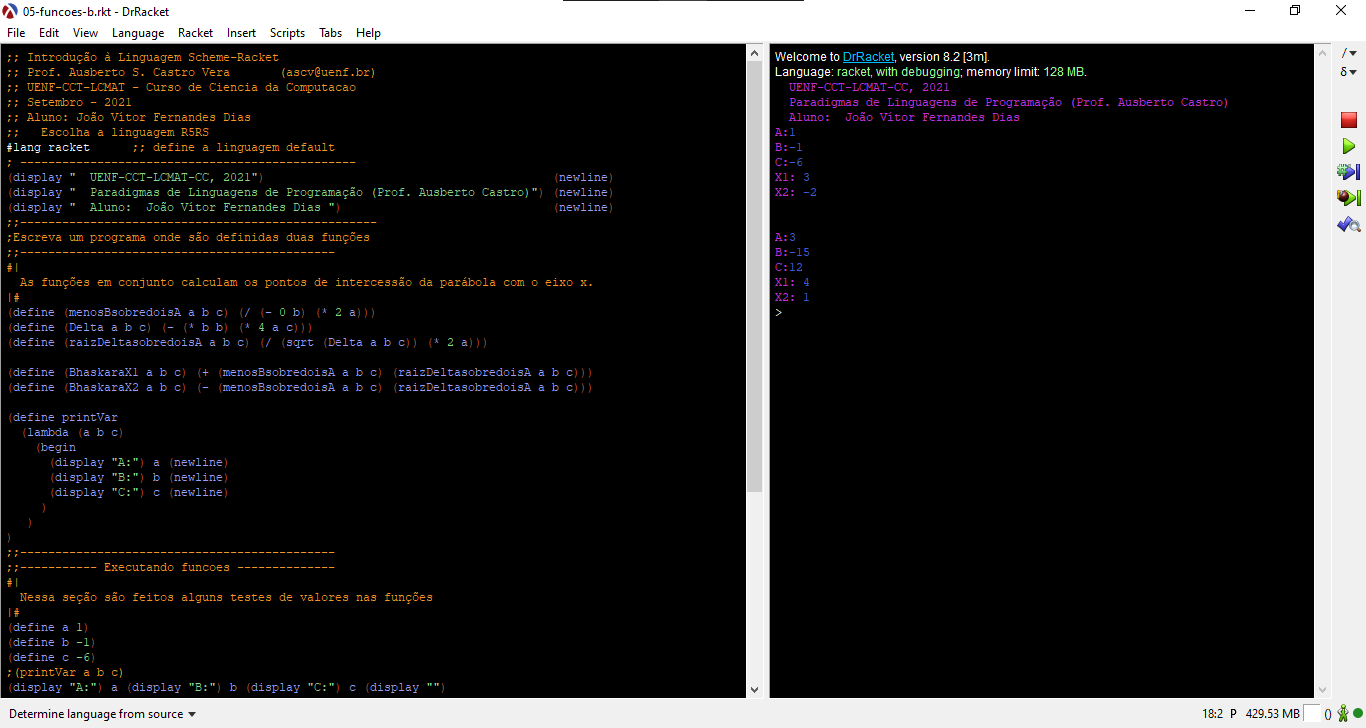
A função “quadrado” recebe o valor de uma variável e retorna o seu valor elevado ao quadrado (ou seja, multiplicado por ele mesmo).

Já as outras, apesar dos diferentes nomes, fazem a mesma coisa. Todas elas recebem o valor de duas variáveis e retornam a soma desses valores.

Já no final do código, são testadas as funções com diferentes valores.

##### Escreva um programa onde são definidas duas funções

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

;; Escolha a linguagem R5RS

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline)

;;---------------------------------------------------

;Escreva um programa onde são definidas duas funções

;;---------------------------------------------

#|

As funções em conjunto calculam os pontos de intercessão da parábola com o eixo x.

|#

(define (menosBsobredoisA a b c) (/ (- 0 b) (\* 2 a)))

(define (Delta a b c) (- (\* b b) (\* 4 a c)))

(define (raizDeltasobredoisA a b c) (/ (sqrt (Delta a b c)) (\* 2 a)))

(define (BhaskaraX1 a b c) (+ (menosBsobredoisA a b c) (raizDeltasobredoisA a b c)))

(define (BhaskaraX2 a b c) (- (menosBsobredoisA a b c) (raizDeltasobredoisA a b c)))

(define printVar

(lambda (a b c)

(begin

(display "A:") a (newline)

(display "B:") b (newline)

(display "C:") c (newline)

)

)

)

;;---------------------------------------------

;;----------- Executando funcoes --------------

#|

Nessa seção são feitos alguns testes de valores nas funções

|#

(define a 1)

(define b -1)

(define c -6)

;(printVar a b c)

(display "A:") a (display "B:") b (display "C:") c (display "")

;(display "-b/2a: ")(menosBsobredoisA a b c)

;(display "b²-4ac: ")(Delta a b c)

;(display "raiz(delta)/2a: ")(raizDeltasobredoisA a b c)

(display "X1: ")(BhaskaraX1 a b c)

(display "X2: ")(BhaskaraX2 a b c)

(newline)(newline)

(define i 3)

(define j -15)

(define k 12)

;(printVar i j k)

(display "A:") i (display "B:") j (display "C:") k (display "")

;(display "-b/2a: ")(menosBsobredoisA a b c)

;(display "b2-4ac: ")(Delta a b c)

;(display "raiz(delta)/2a: ")(raizDeltasobredoisA a b c)

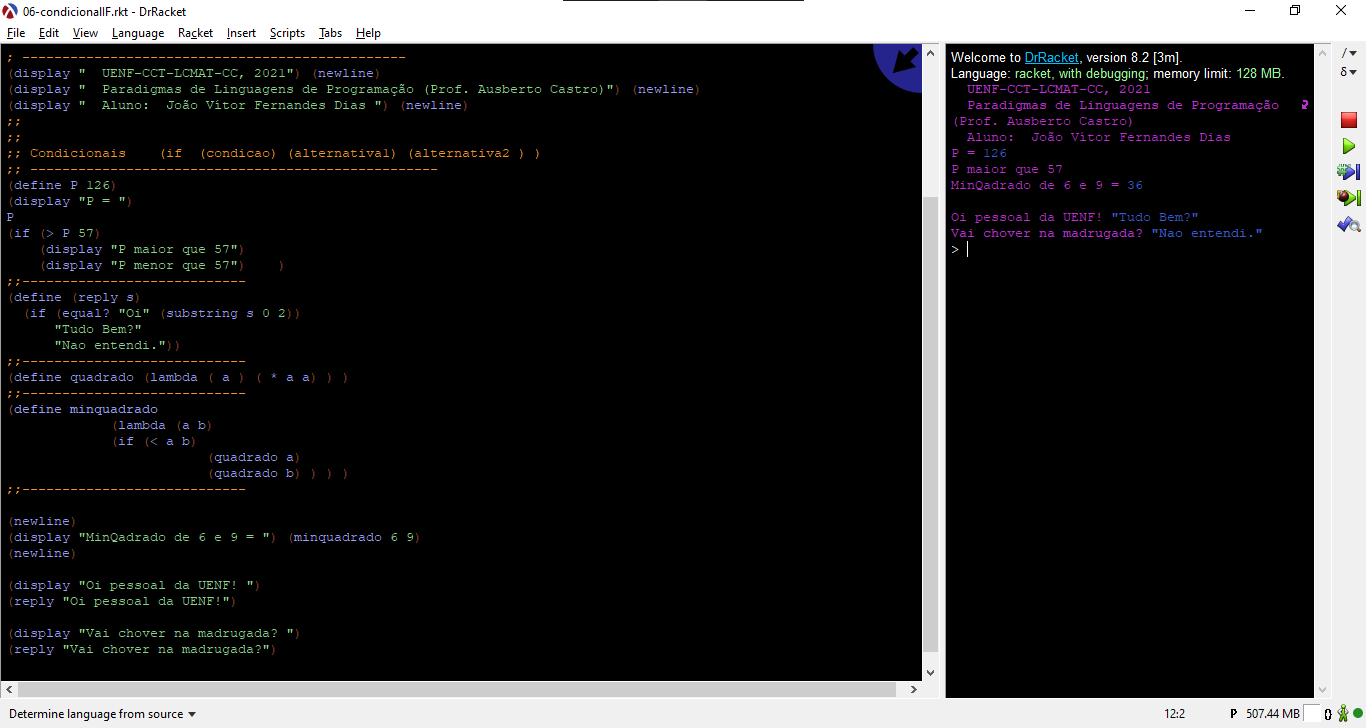
(display "X1: ")(BhaskaraX1 i j k)

(display "X2: ")(BhaskaraX2 i j k)

#### Arquivo 06-condicionalIF.rtk 06/09/21: 15h34 – 16h32

##### Executar e explicar o programa

###### Print



###### Explicação

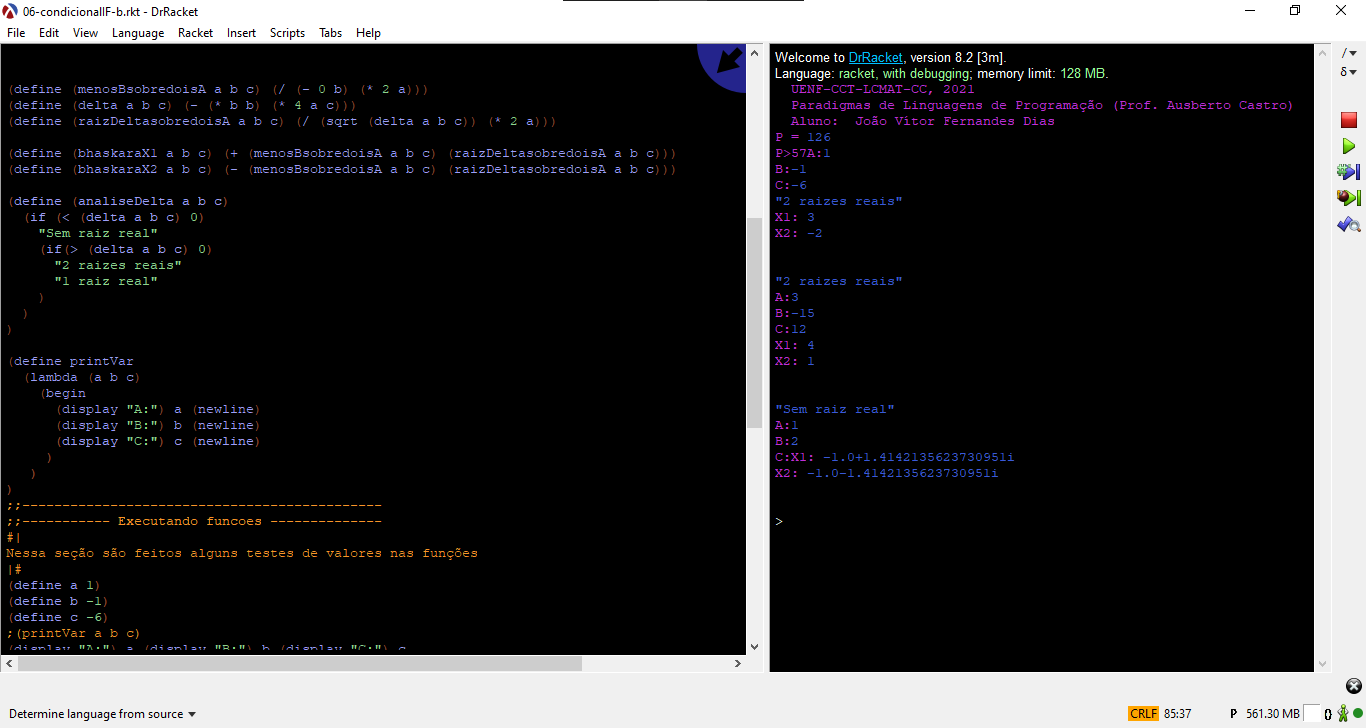
De uma forma geral, o programa exemplifica o uso da estrutura condicional if.

Mais especificamente ele inicia comparando o valor de uma variável e retornando textos diferentes de acordo com o seu valor. Mais abaixo define uma função chamada reply que responde “tudo bem” caso a substring indo do índice 0 ao 2 for igual a “Oi”, e caso não seja, responde “Não entendi”. Outras duas funções que trabalham em conjunto são definidas abaixo. Uma que retorna o menor valor entre dois números elevados ao quadrado e outra que calcula o valor de um número elevado ao quadrado.

Por fim, as funções são testadas.

##### Escreva um programa com dois condicionais

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

;; Escolha a linguagem R5RS

;; Ajuda: http://docs.racket-lang.org/guide/syntax-overview.html#(part.\_.Conditionals\_with\_if\_\_and\_\_or\_\_and\_cond)

#lang racket ;; define a linguagem default

; -----------Condicionais---------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline)

(define (menosBsobredoisA a b c) (/ (- 0 b) (\* 2 a)))

(define (delta a b c) (- (\* b b) (\* 4 a c)))

(define (raizDeltasobredoisA a b c) (/ (sqrt (delta a b c)) (\* 2 a)))

(define (bhaskaraX1 a b c) (+ (menosBsobredoisA a b c) (raizDeltasobredoisA a b c)))

(define (bhaskaraX2 a b c) (- (menosBsobredoisA a b c) (raizDeltasobredoisA a b c)))

(define (analiseDelta a b c)

(if (< (delta a b c) 0)

"Sem raiz real"

(if(> (delta a b c) 0)

"2 raizes reais"

"1 raiz real")))

(define (execucao a b c) (newline)

(display "A:") (display a) (display " B:") (display b) (display " C:") (display c)

(display " ")

(display (analiseDelta a b c)) (newline)

(display "X1: ") (display (bhaskaraX1 a b c))

(display " X2: ") (display (bhaskaraX2 a b c)) (newline) )

;;----------- Executando funcoes --------------

; Nessa seção são feitos alguns testes de valores nas funções

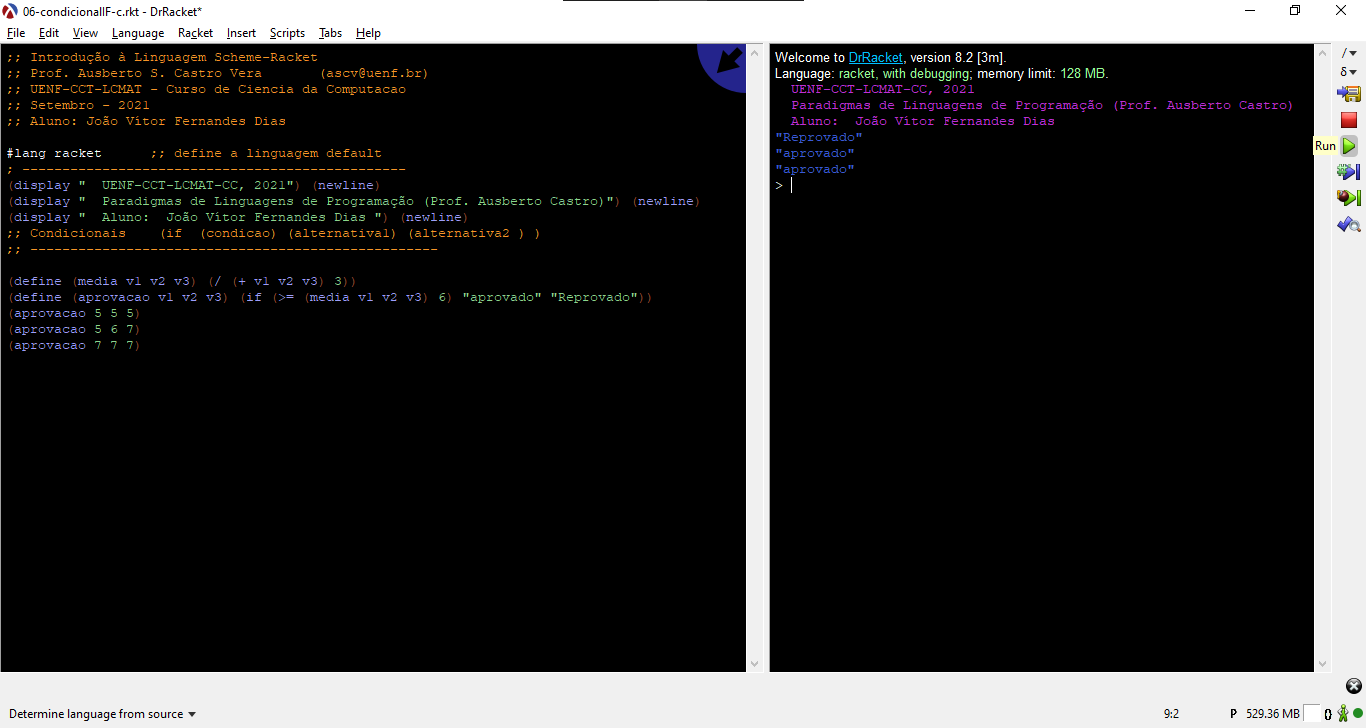
(execucao 1 -1 -6)

(execucao 3 -15 12)

(execucao 1 2 3)

##### Escreva um programa para calcular a média de três notas e indique “Aprovado” se for maior ou igual a 6,0, e “Reprovado”, caso contrário

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline)

;; Condicionais (if (condicao) (alternativa1) (alternativa2 ) )

;; ---------------------------------------------------

(define (media v1 v2 v3) (/ (+ v1 v2 v3) 3))

(define (aprovacao v1 v2 v3) (if (>= (media v1 v2 v3) 6) "aprovado" "Reprovado"))

(aprovacao 5 5 5)

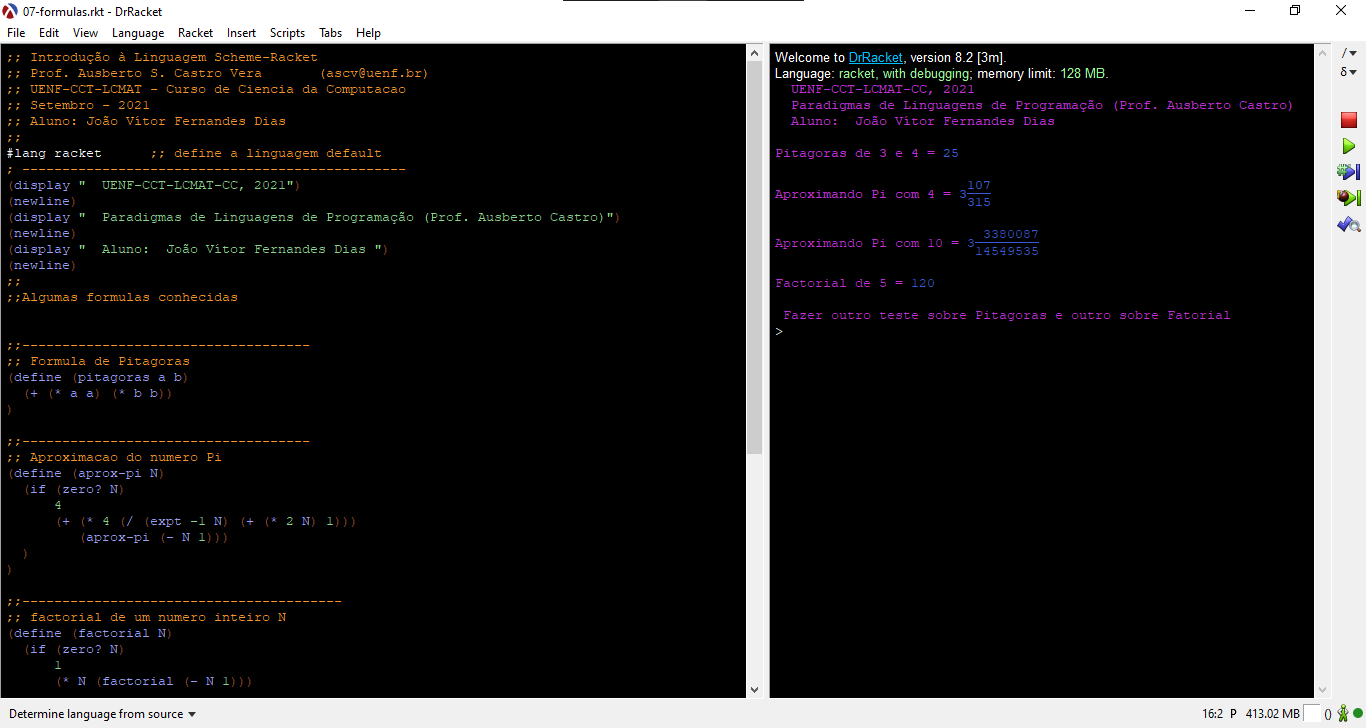
(aprovacao 5 6 7)

(aprovacao 7 7 7)

#### Arquivo 07-formulas.rtk 16h34 – 17h21 Dúvida

##### Executar e explicar o programa - **DÚVIDA**

###### Print



###### Explicação

São definidas três funções:

A primeira retorna a soma dos quadrados de dois números (a\*a+b\*b)

A segunda faz uma recursividade de tamanho N para encontrar um valor próximo do PI (Não entendi como funciona matematicamente - DÚVIDA)

A terceira calcula recursivamente o valor fatorial

E por fim são testadas essas funções com alguns valores de teste.

##### Escreva um programa que calcule o fatorial de um número de uma forma diferente da apresentada.

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

;;

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline)(newline)

;;Algumas formulas conhecidas

;;------------------------------------

;; Formula de Pitagoras

(define (pitagoras a b) (+ (\* a a) (\* b b)) )

;;------------------------------------

;; Aproximacao do numero Pi

(define (aprox-pi N)

(if (zero? N)

4

(+ (\* 4 (/ (expt -1 N) (+ (\* 2 N) 1))) (aprox-pi (- N 1)))

; 4\*(-1^N) /(2\*N+1) + aprox-pi(N-1)

)

)

;;----------------------------------------

;; factorial de um numero inteiro N

(define (factorial N)

(if (zero? N)

1

(\* N (factorial (- N 1)))

)

)

;;----------------------------------------

;; fatorial de um numero inteiro N

(define (fatorialR cont N)

(if (<= cont N)

(\* cont (fatorialR (+ cont 1) N))

1

)

)

(define (fatorial N)

(fatorialR 1 N)

)

(display "Pitagoras de 3 e 4 = ") (pitagoras 3 4) (newline)

(display "Aproximando Pi com 4 = ") (aprox-pi 4) (newline)

(display "Aproximando Pi com 10 = ") (aprox-pi 10) (newline)

(display "Factorial de 5 = ") (factorial 5) (newline)

(display " Fazer outro teste sobre Pitagoras e outro sobre Fatorial") (newline)

(display "Pitagoras de 6 e 8 = ") (pitagoras 6 8)

(display "Factorial de 6 = ") (factorial 6) (newline)

(newline)(newline)

(display "Fatorial JV 5 = ") (fatorial 5) (newline)

(display "Fatorial JV 6 = ") (fatorial 6) (newline)

#### Arquivo 08-condicional.rtk

##### Execute o programa e indique o que faz

###### Print



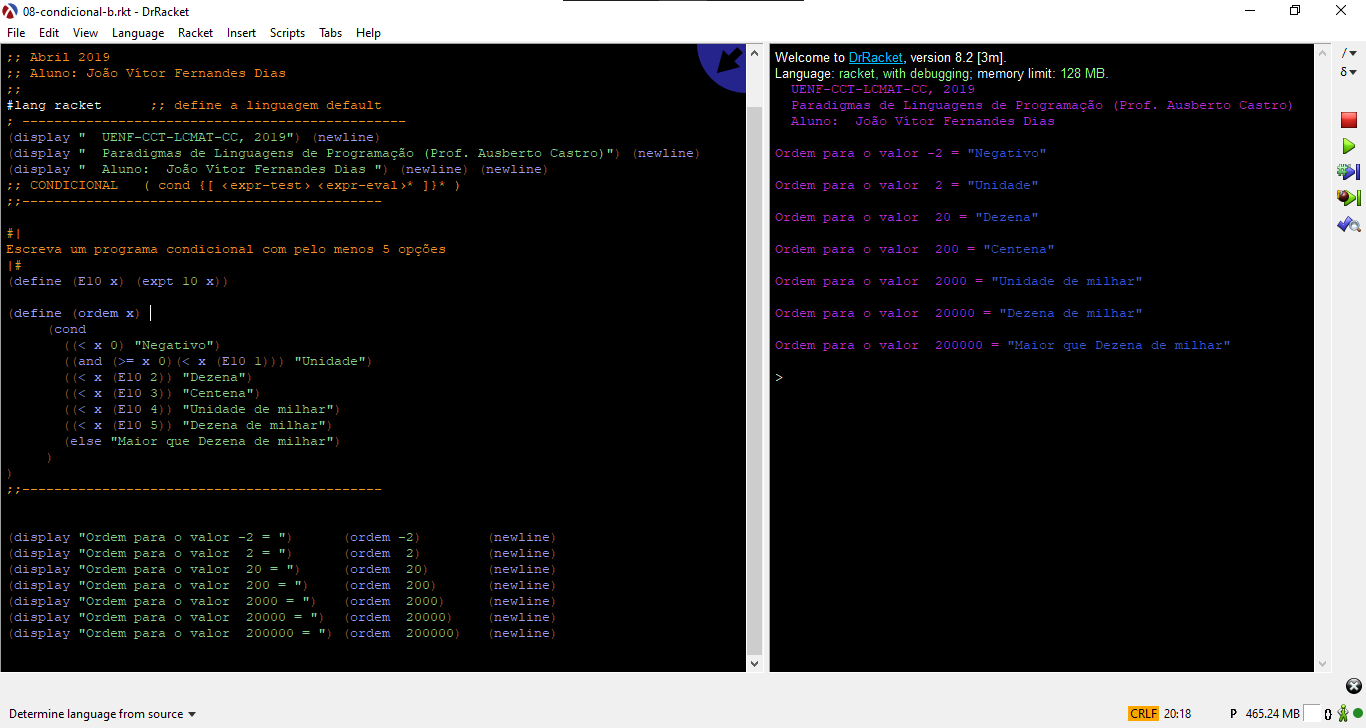
###### Explicação

O programa ilustra o uso da estrutura condicional-cond.

Ele funciona de maneira similar à estrutura switch...case da linguagem C, onde, baseado em uma variável específica (nesse caso a variável é “quantidade”, da função “taxa”) estabelece-se diversos casos possíveis, cada um com uma execução de funções ou retorno de valores específicos. Outra forma de explicar seria com a ideia de encadeamento de if’s, onde pode-se colocar várias condicionais inseridas uma na outra a fim de haver várias alternativas possíveis, porém com a estrutura cond, isso se torna mais denso sintaticamente.

##### Escreva um programa condicional com pelo menos 5 opções

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Abril 2019

;; Aluno: João Vítor Fernandes Dias

;;

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2019") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

;; CONDICIONAL ( cond {[ ‹expr-test› ‹expr-eval›\* ]}\* )

;;---------------------------------------------

#|

Escreva um programa condicional com pelo menos 5 opções

|#

(define (E10 x) (expt 10 x))

(define (ordem x)

(cond

((< x 0) "Negativo")

((and (>= x 0)(< x (E10 1))) "Unidade")

((< x (E10 2)) "Dezena")

((< x (E10 3)) "Centena")

((< x (E10 4)) "Unidade de milhar")

((< x (E10 5)) "Dezena de milhar")

(else "Maior que Dezena de milhar")

)

)

;;---------------------------------------------

(display "Ordem para o valor -2 = ") (ordem -2) (newline)

(display "Ordem para o valor 2 = ") (ordem 2) (newline)

(display "Ordem para o valor 20 = ") (ordem 20) (newline)

(display "Ordem para o valor 200 = ") (ordem 200) (newline)

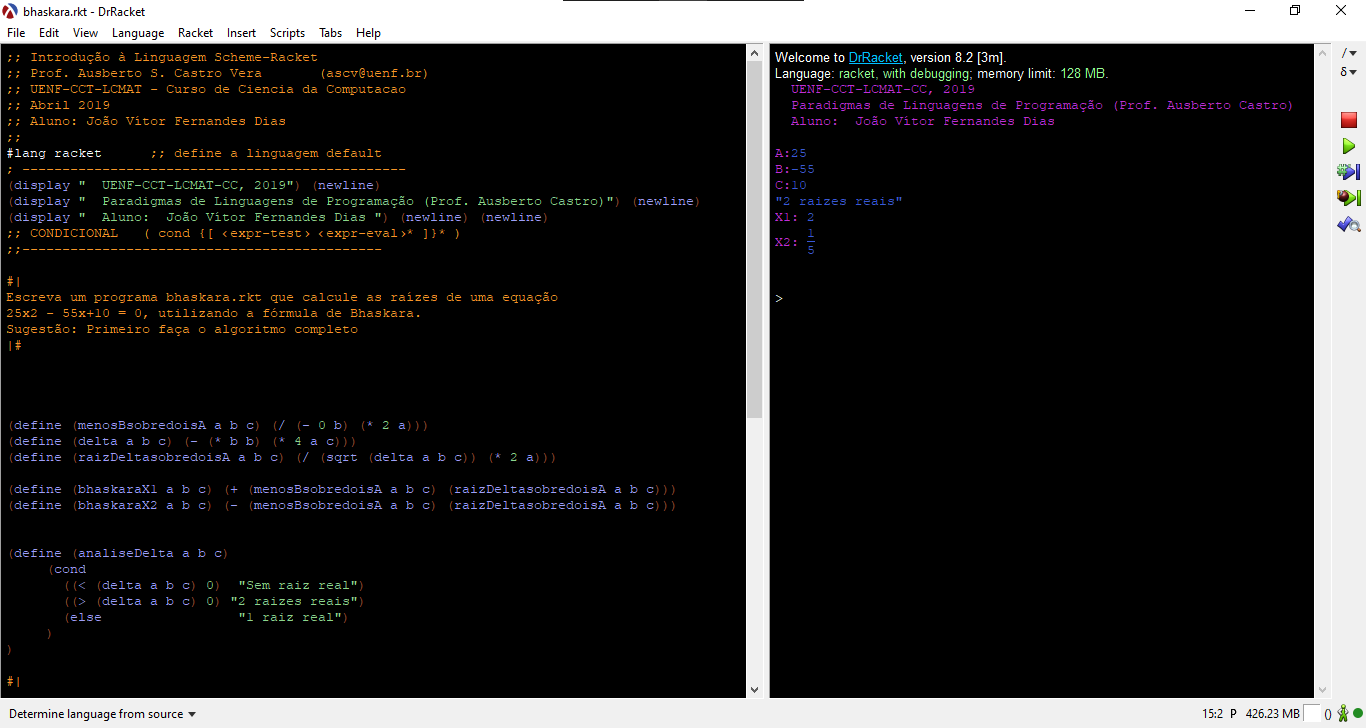
(display "Ordem para o valor 2000 = ") (ordem 2000) (newline)

(display "Ordem para o valor 20000 = ") (ordem 20000) (newline)

(display "Ordem para o valor 200000 = ") (ordem 200000) (newline)

##### Escreva um programa bhaskara.rkt que calcule as raízes de uma equação 25x2 - 55x+10 = 0, utilizando a fórmula de Bhaskara. Sugestão: Primeiro faça o algoritmo completo

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Abril 2019

;; Aluno: João Vítor Fernandes Dias

;;

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2019") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

;; CONDICIONAL ( cond {[ ‹expr-test› ‹expr-eval›\* ]}\* )

;;---------------------------------------------

#|

Escreva um programa bhaskara.rkt que calcule as raízes de uma equação

25x2 - 55x+10 = 0, utilizando a fórmula de Bhaskara.

Sugestão: Primeiro faça o algoritmo completo

|#

(define (menosBsobredoisA a b c) (/ (- 0 b) (\* 2 a)))

(define (delta a b c) (- (\* b b) (\* 4 a c)))

(define (raizDeltasobredoisA a b c) (/ (sqrt (delta a b c)) (\* 2 a)))

(define (bhaskaraX1 a b c) (+ (menosBsobredoisA a b c) (raizDeltasobredoisA a b c)))

(define (bhaskaraX2 a b c) (- (menosBsobredoisA a b c) (raizDeltasobredoisA a b c)))

(define (analiseDelta a b c)

(cond

((< (delta a b c) 0) "Sem raiz real")

((> (delta a b c) 0) "2 raizes reais")

(else "1 raiz real")

)

)

#|

(define (analiseDelta a b c)

(if (< (delta a b c) 0)

"Sem raiz real"

(if(> (delta a b c) 0)

"2 raizes reais"

"1 raiz real"

)

)

)

|#

;;---------------------------------------------

;;----------- Executando funcoes --------------

#|

Nessa seção são feitos alguns testes de valores nas funções

|#

(define a 25)

(define b -55)

(define c 10)

;(printVar a b c)

(display "A:") a (display "B:") b (display "C:") c

;(display "-b/2a: ")(menosBsobredoisA a b c)

;(display "b²-4ac: ")(Delta a b c)

;(display "raiz(delta)/2a: ")(raizDeltasobredoisA a b c)

(analiseDelta a b c)

(display "X1: ")(bhaskaraX1 a b c)

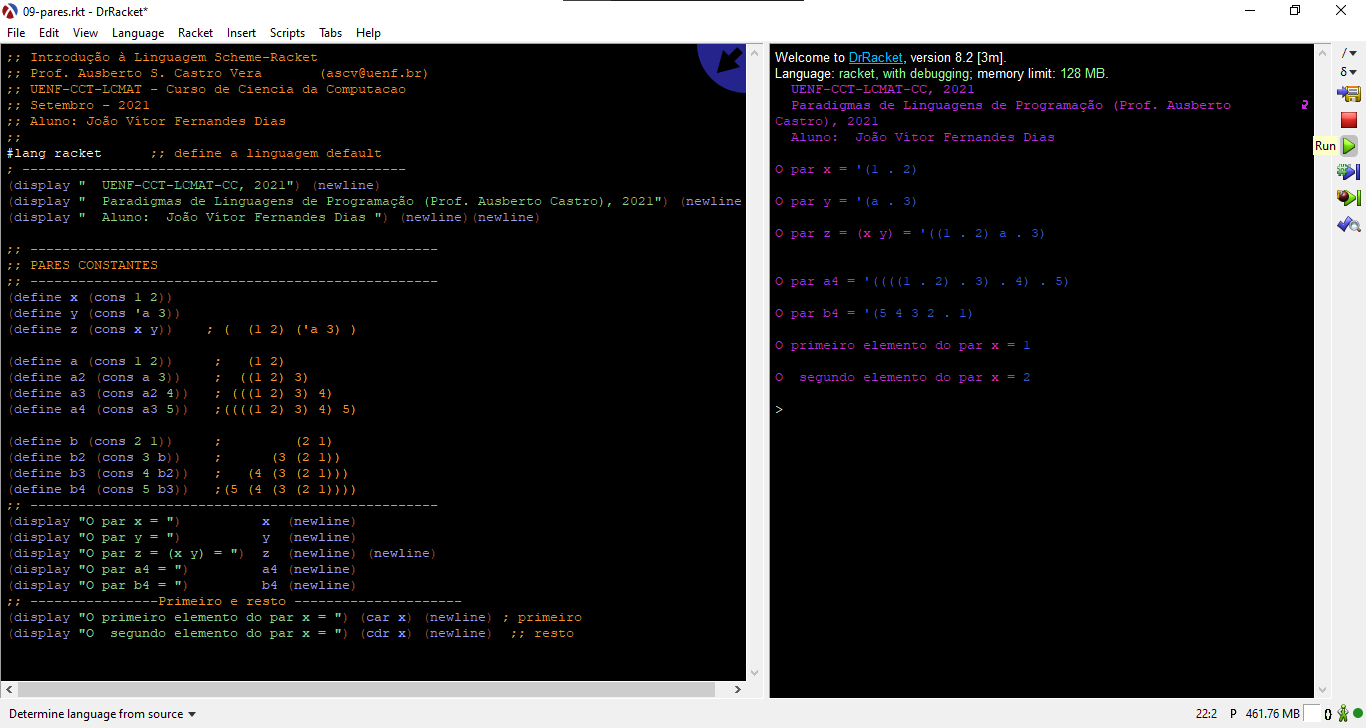
(display "X2: ")(bhaskaraX2 a b c)

(newline)(newline)

#### Arquivo 09-pares.rtk 07/09/21 15h21 – 15h32

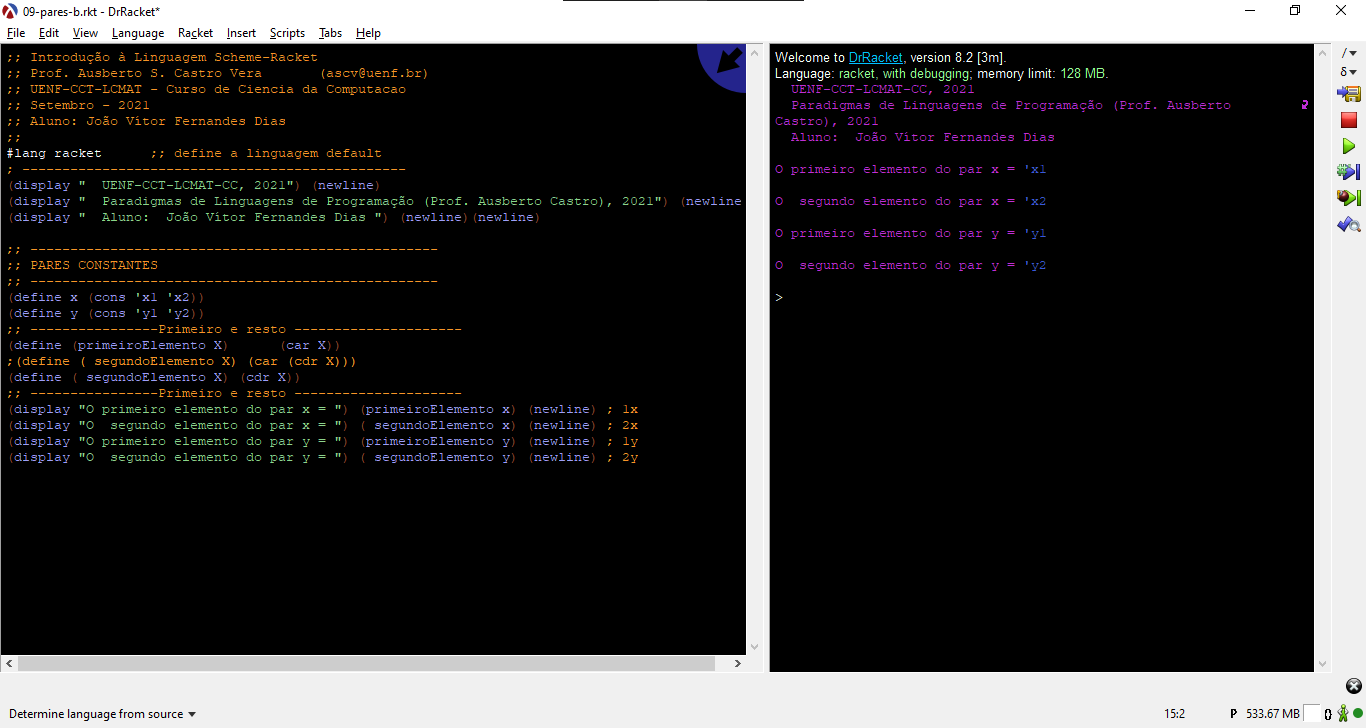
##### Execute o programa 15h21

###### Print



##### Escreva um NOVO programa para construir dois pares e indicar em cada um deles o primeiro e o segundo elemento 15h22 – 15h32

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

;;

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro), 2021") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline)(newline)

;; ---------------------------------------------------

;; PARES CONSTANTES

;; ---------------------------------------------------

(define x (cons 'x1 'x2))

(define y (cons 'y1 'y2))

;; ----------------Primeiro e resto ---------------------

(define (primeiroElemento X) (car X))

;(define ( segundoElemento X) (car (cdr X)))

(define ( segundoElemento X) (cdr X))

;; ----------------Primeiro e resto ---------------------

(display "O primeiro elemento do par x = ") (primeiroElemento x) (newline) ; 1x

(display "O segundo elemento do par x = ") ( segundoElemento x) (newline) ; 2x

(display "O primeiro elemento do par y = ") (primeiroElemento y) (newline) ; 1y

(display "O segundo elemento do par y = ") ( segundoElemento y) (newline) ; 2y

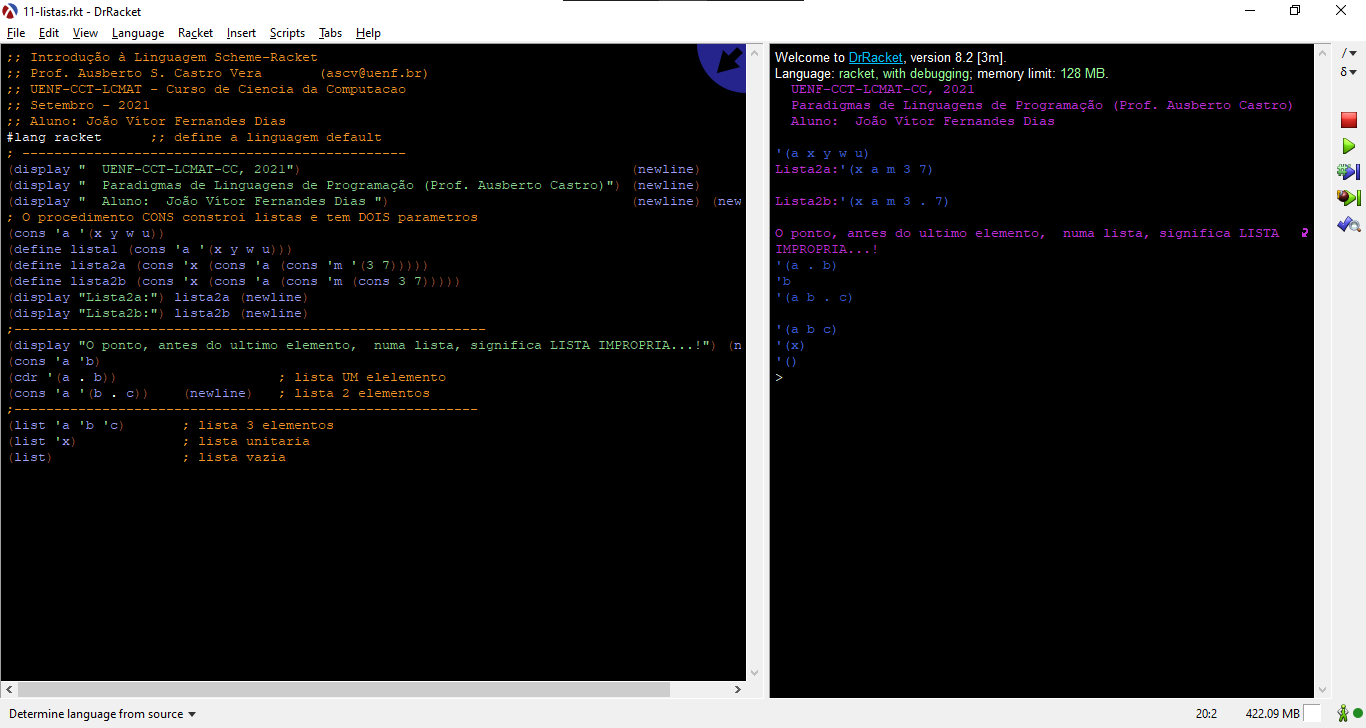
#### Arquivo 10-listas.rtk e 11-listas.rtk 15h56 – 17h51

##### Executar os programas e observe quantos métodos existem para construir listas 15h56 – 16h24

###### Print 10

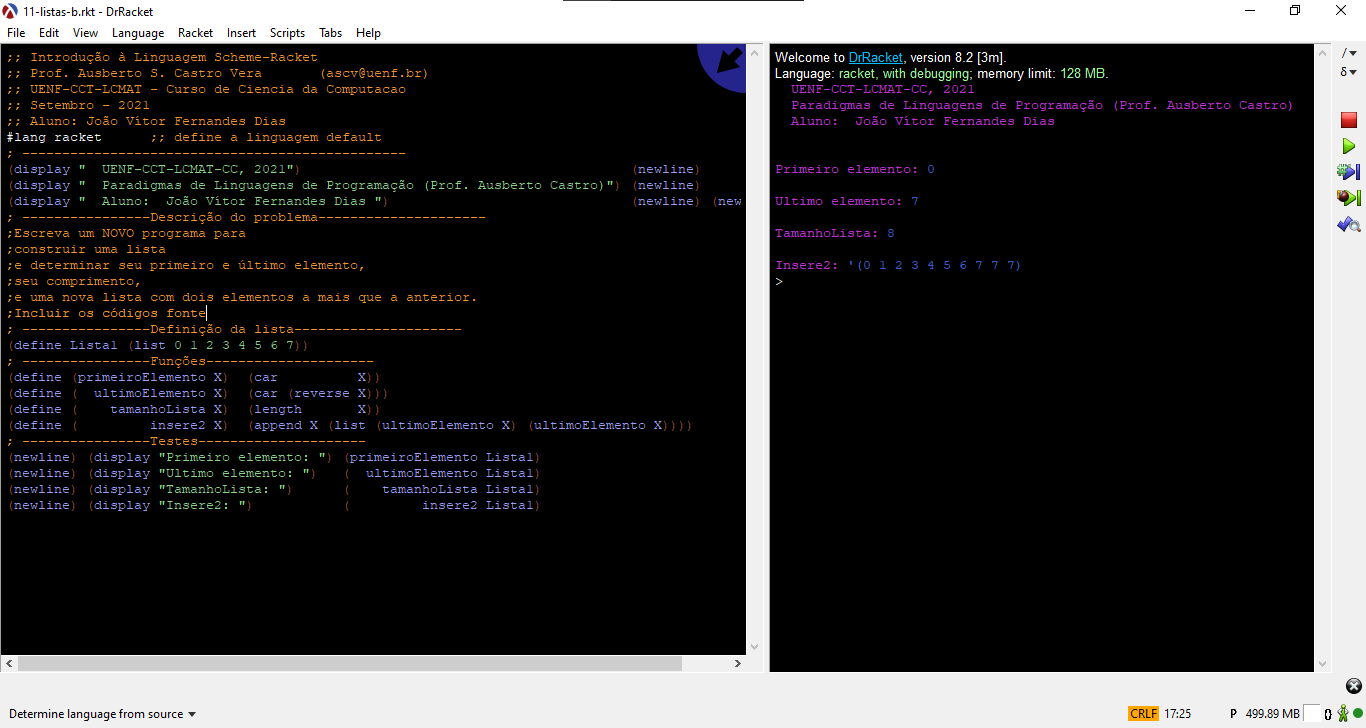


###### Print 11



##### Escreva um NOVO programa para construir uma lista e determinar seu primeiro e último elemento, seu comprimento, e uma nova lista com dois elementos a mais que a anterior. Incluir os códigos fonte 16h33 – 17h16

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

; ----------------Descrição do problema---------------------

;Escreva um NOVO programa para

;construir uma lista

;e determinar seu primeiro e último elemento,

;seu comprimento,

;e uma nova lista com dois elementos a mais que a anterior.

;Incluir os códigos fonte

; ----------------Definição da lista---------------------

(define Lista1 (list 0 1 2 3 4 5 6 7))

; ----------------Funções---------------------

(define (primeiroElemento X) (car X))

(define ( ultimoElemento X) (car (reverse X)))

(define ( tamanhoLista X) (length X))

(define ( insere2 X) (append X (list (ultimoElemento X) (ultimoElemento X))))

; ----------------Testes---------------------

(newline) (display "Primeiro elemento: ") (primeiroElemento Lista1)

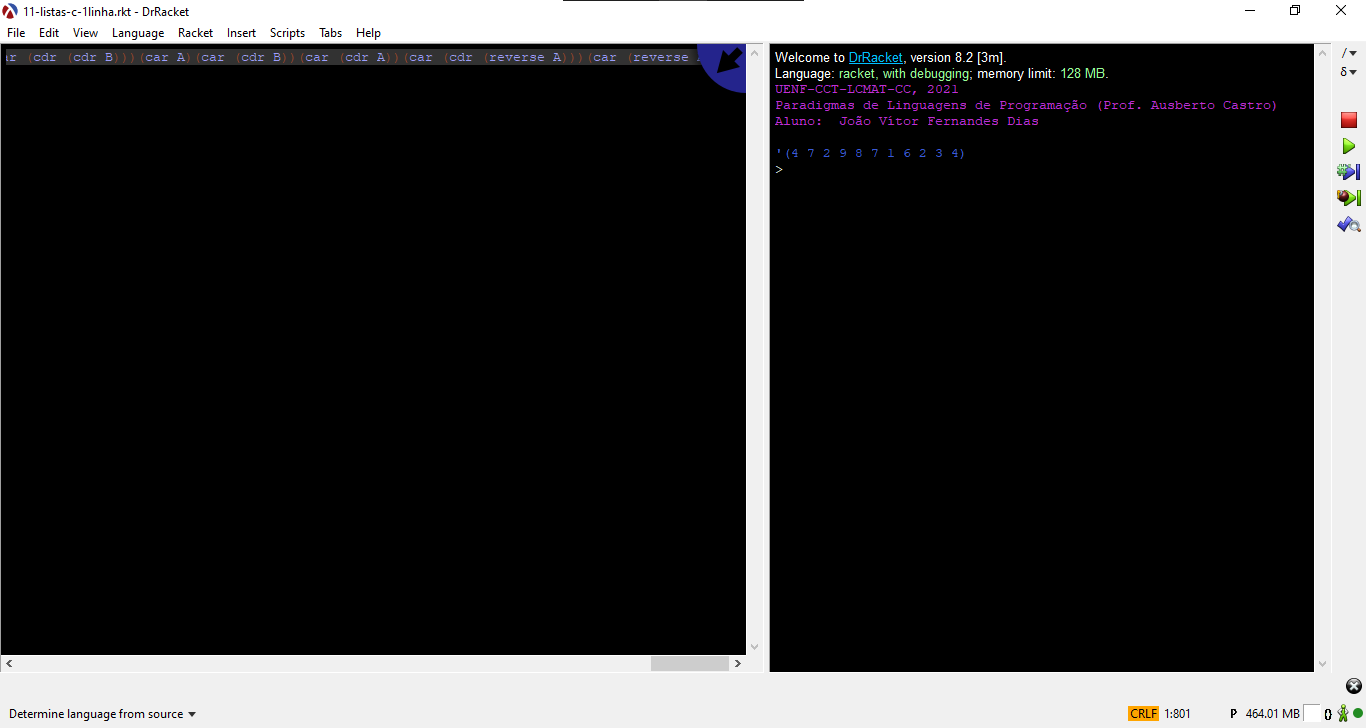
(newline) (display "Ultimo elemento: ") ( ultimoElemento Lista1)

(newline) (display "TamanhoLista: ") ( tamanhoLista Lista1)

(newline) (display "Insere2: ") ( insere2 Lista1)

##### Utilizando uma ÚNICA linha de comandos, escreva um NOVO programa Racket para construir a lista (4 7 2 9 8 7 1 6 2 3 4) a partir das listas A= (1 2 3 4) e B= (5 6 7 8 9) 17h16 – 17h51

###### Print

**

###### Código

#|Introdução à Linguagem Scheme-Racket Prof. Ausberto S. Castro Vera (ascv@uenf.br)UENF-CCT-LCMAT - Curso de Ciencia da ComputacaoSetembro - 2021Aluno: João Vítor Fernandes Dias|# #lang racket (display "UENF-CCT-LCMAT-CC, 2021")(newline)(display "Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)")(newline)(display "Aluno: João Vítor Fernandes Dias ")(newline) (newline)#| Utilizando uma ÚNICA linha de comandos, escreva um NOVO programa Racket para construir a lista (4 7 2 9 8 7 1 6 2 3 4) a partir das listas A= (1 2 3 4) e B= (5 6 7 8 9) |#(let ((A (list 1 2 3 4)) (B (list 5 6 7 8 9)))(list(car (reverse A))(car (cdr (cdr B)))(car (cdr A))(car (reverse B))(car (cdr (reverse B)))(car (cdr (cdr B)))(car A)(car (cdr B))(car (cdr A))(car (cdr (reverse A)))(car (reverse A))))

**A função especificamente:**

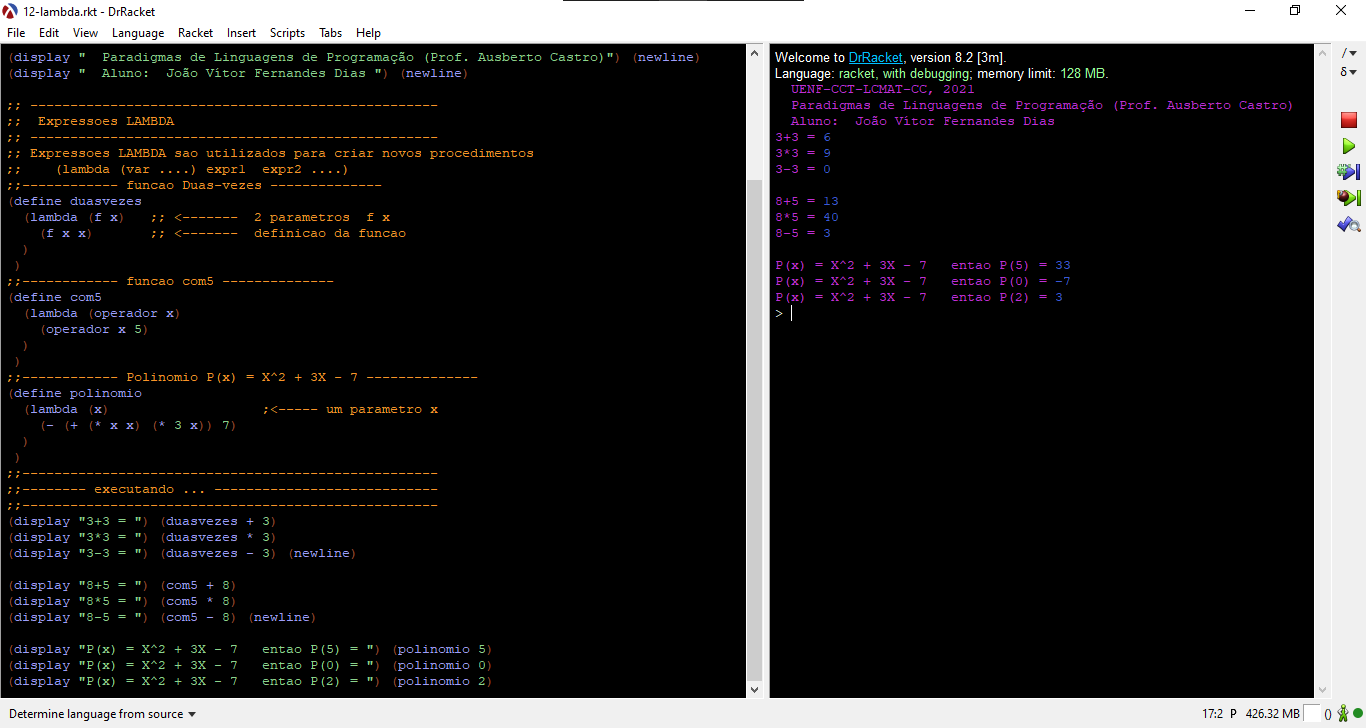
(let ((A (list 1 2 3 4)) (B (list 5 6 7 8 9)))(list(car (reverse A))(car (cdr (cdr B)))(car (cdr A))(car (reverse B))(car (cdr (reverse B)))(car (cdr (cdr B)))(car A)(car (cdr B))(car (cdr A))(car (cdr (reverse A)))(car (reverse A))))

#### Arquivo 12-lambda.rtk e 13-lambda.rtk 19h34 – 20h51

##### Execute os programas e indique o que faz cada um deles 19h34 – 20h09

###### Arquivo 12-lambda.rtk 19h34 – 19h43

Print



Explicação

Este programa ilustra o uso do lambda (embora eu não tenha entendido ainda qual a necessidade do uso dele ao invés do uso padrão do define).

É criado uma função chamada “duasvezes” que faz uma devida operação de um valor com ele mesmo. Tanto a operação, quanto o valor são enviados como parâmetros.

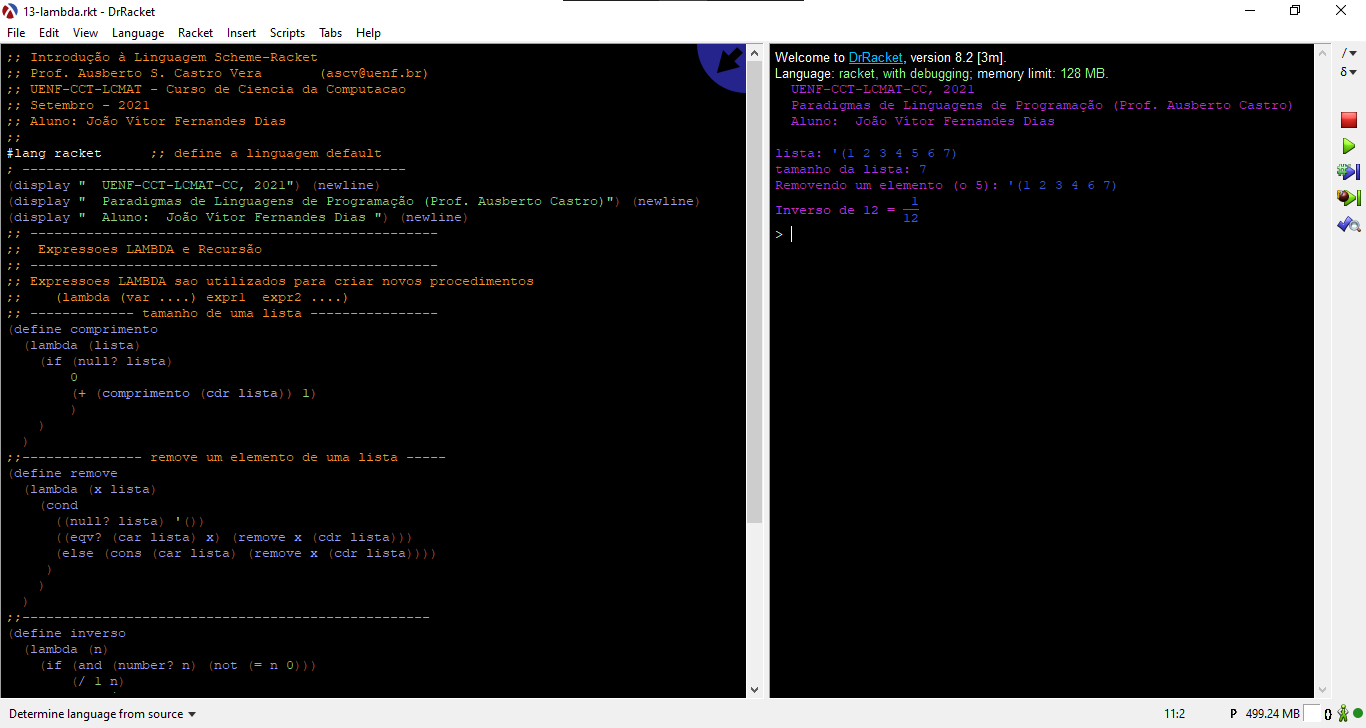
Em seguida é criada uma função “com5” que faz uma devida operação de um valor com o 5, sendo sempre o 5 o segundo termo da operação. Tanto a operação, quanto o valor são enviados como parâmetros.

A próxima função definida é a função “polinomio” que retorna o valor da operação “x²+3x-7”, sendo x uma variável que foi enviada como parâmetro.

Por fim, são testadas as funções com diversos valores e operações diferentes.

###### Arquivo 13-lambda.rtk 19h44 – 20h09

Print



Explicação

A primeira função calcula o comprimento da lista (assim como o length) de forma recursiva. Enquanto a lista não estiver vazia, somará 1 e chamará novamente a função, enviando todos os elementos, menos o primeiro, como parâmetro. Quando estiver vazia, retornará 0.

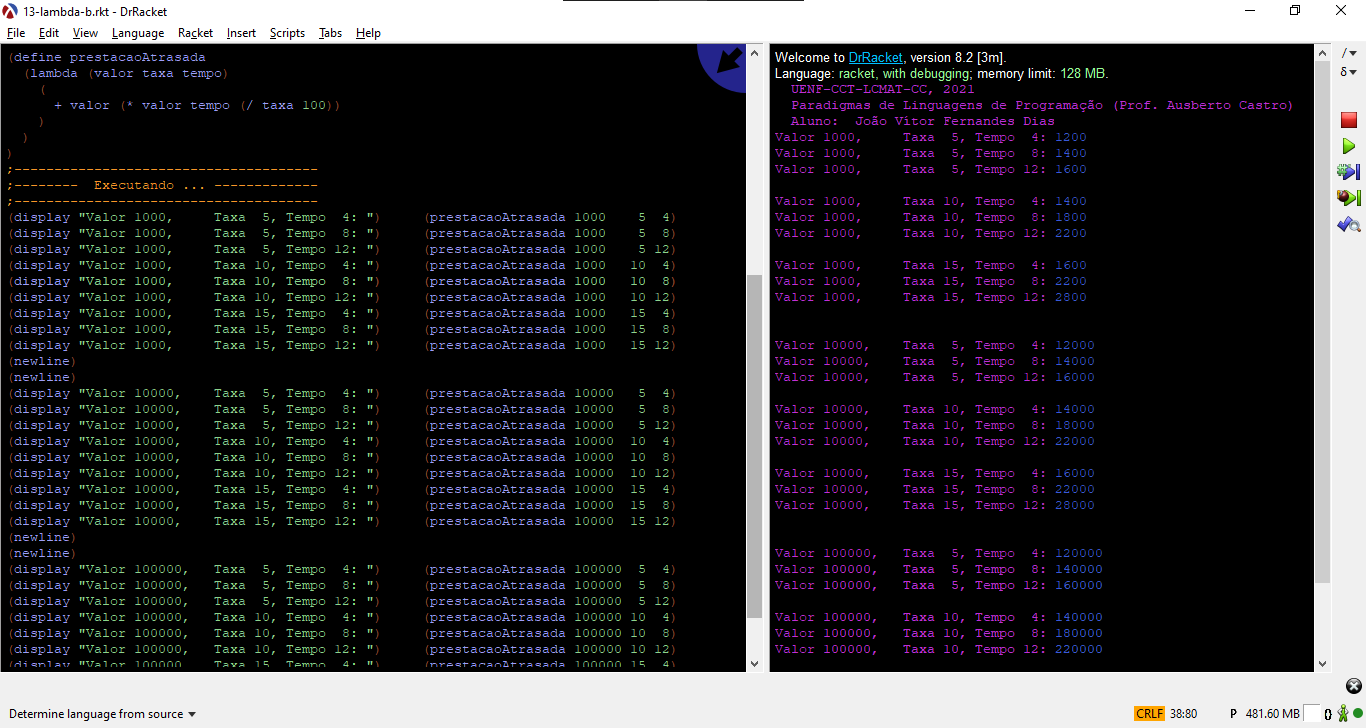
A segunda função faz a busca na lista e quando o encontrar, removerá (só que de forma mais complexa e recursiva). Ele recebe como parâmetro o valor que está sendo buscado e a lista, então entra na condicional: caso a lista estiver vazia, retornará uma lista vazia. Caso o primeiro elemento da lista for equivalente ao valor que está sendo buscado, chama-se novamente a função enviando como parâmetro toda a lista após o primeiro elemento (é dessa forma que o elemento é removido). Caso não seja nenhum dos casos acima, ele adiciona o primeiro elemento da lista ao retorno da chamada recursiva da função “remove” que envia o valor a ser removido e todo o resto da lista.

A terceira função, chamada “inverso” confere se o valor n recebido é um número e se ele não é igual a zero, se ambas as condições forem verdadeiras, ele retornara o valor referente ao inverso no número n, ou seja, 1/n.

Abaixo é criada a lista e testadas as funções descritas acima com valores de teste.

##### Crie um procedimento para realizar o cálculo de uma prestação em atraso, utilizando a fórmula Prest = valor + (valor\*(taxa/100) \*tempo). Dar exemplos. 20h09 – 20h23]

###### Print



##### O que faz o seguinte procedimento abcd: 20h23 – 20h51

###### Procedimento

(define abcd

(lambda (n)

(let f ((i 2))

(cond

((>= i n) '())

((integer? (/ n i))

(cons i (f (+ i 1))))

(else (f (+ i 1)))))))

###### Print



###### Explicação

Após alguns testes que são descritos de forma menos formal abaixo, pode-se compreender que este código retorna uma lista contendo todos os múltiplos de um determinado valor n, excluindo os valores 1 e o próprio valor n.

###### Código

(define abcd

(lambda (n) ;recebe um valor n

(let f ((i 2)) ;dentro desse escopo torna o valor i igual a 2

;Porém não sei qual a função do f nesse contexto

;(

; Após rodar o código, entendi que o é uma função que está

; sendo definida e está sendo utilizada na recursão,

; utilizando o i como contador que vai de 2 a n, porém,

; ainda não entendi de que forma este f está sendo definido

;)

(cond ;ocorre uma condicional

((>= i n) '()) ;caso i (que equivale a 2) seja maior ou igual ao valor

;recebido, retorna uma lista vazia

((integer? (/ n i)) ;caso n/i seja inteiro,

(cons i (f (+ i 1))) ;cria um par contendo i e o resultado de uma função f

) ;que envia como parâmetro o valor i+1

(else (f (+ i 1))) ;caso não seja nenhuma das situações acima, chama a função f

) ;que envia como parâmetro o valor i+1

)

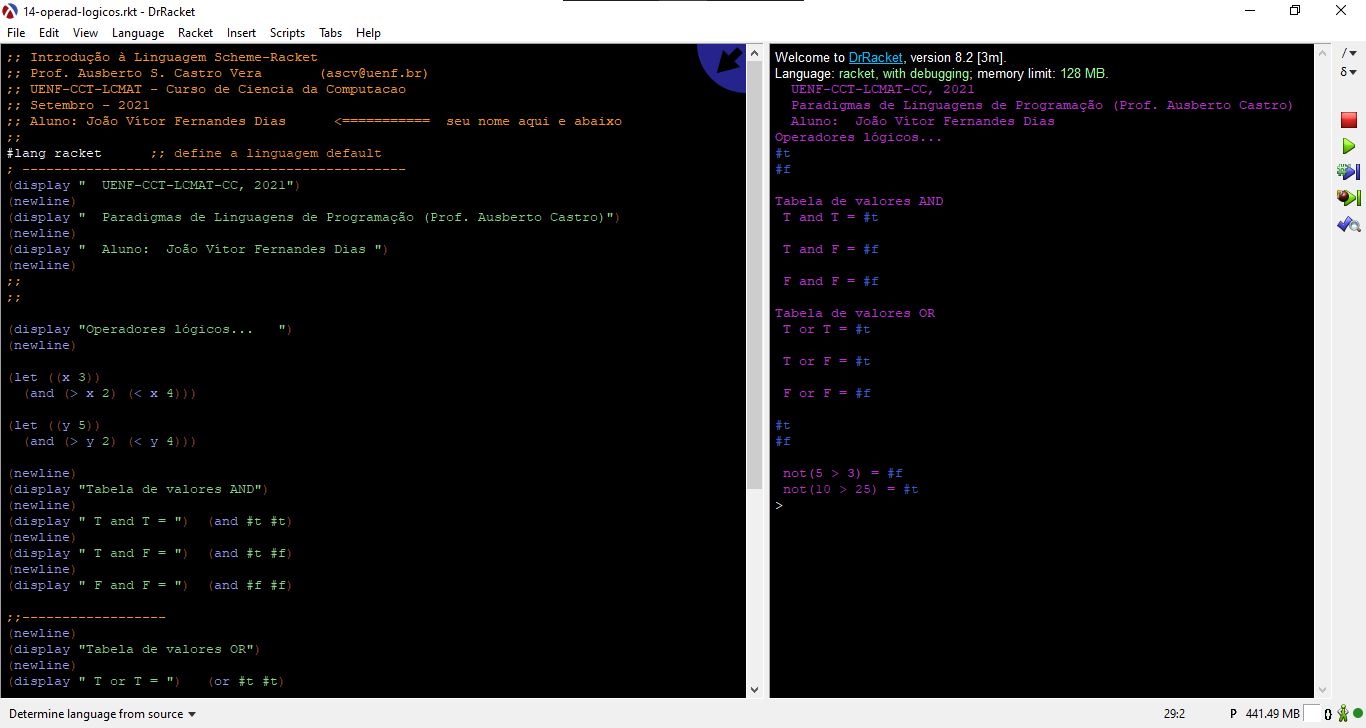
)

)

#### Arquivo 14-operad-logicos.rtk Operadores lógicos 21h00 – 21h28

##### Executar o programa e indicar o seu conteúdo 21h00 – 21h18

###### Print

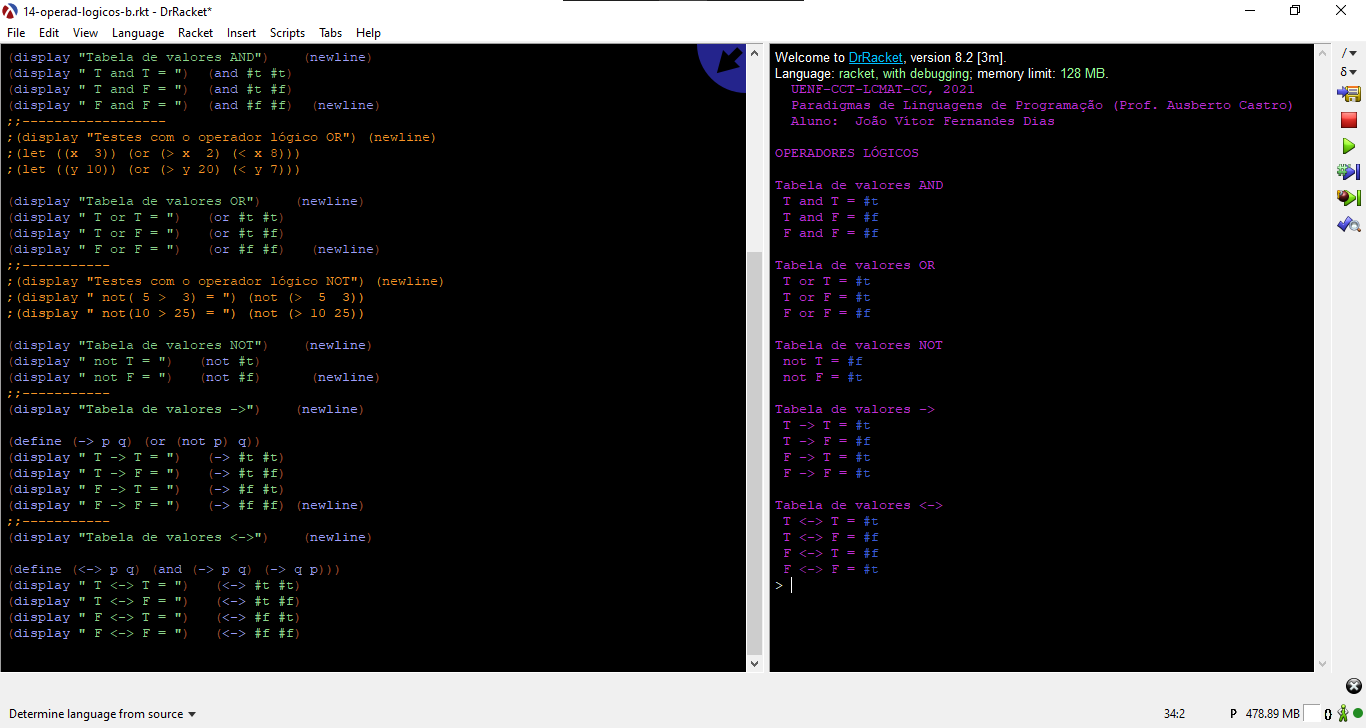


###### Explicação

Nesse código é demonstrado o uso dos operadores lógicos AND, OR e NOT em Racket. Começa fazendo alguns testes com a expressão let para que, usando o operador AND sejam retornados os valores #t e #f (#t equivale ao valor true (verdadeiro) e #f equivale ao valor false (falso)), posteriormente é mostrada a tabela referente aos valores da tabela-verdade do operador AND, o mesmo se repete para o operador OR (porém, mostrando primeiro a tabela e só depois fazendo os testes com a expressão let). Por fim, são testadas duas expressões lógicas utilizando o operador OR, a primeira retornando #f e a segunda #t.

##### Escreva e teste pelo menos cinco operações lógicas 21h18 – 21h28

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

; ------------------------------------------------

(display "OPERADORES LÓGICOS") (newline) (newline)

;(display "Testes com o operador lógico AND") (newline)

;(let ((x 3)) (and (> x 2) (< x 4)))

;(let ((y 5)) (and (> y 2) (< y 4)))

(display "Tabela de valores AND") (newline)

(display " T and T = ") (and #t #t)

(display " T and F = ") (and #t #f)

(display " F and F = ") (and #f #f) (newline)

;;------------------

;(display "Testes com o operador lógico OR") (newline)

;(let ((x 3)) (or (> x 2) (< x 8)))

;(let ((y 10)) (or (> y 20) (< y 7)))

(display "Tabela de valores OR") (newline)

(display " T or T = ") (or #t #t)

(display " T or F = ") (or #t #f)

(display " F or F = ") (or #f #f) (newline)

;;-----------

;(display "Testes com o operador lógico NOT") (newline)

;(display " not( 5 > 3) = ") (not (> 5 3))

;(display " not(10 > 25) = ") (not (> 10 25))

(display "Tabela de valores NOT") (newline)

(display " not T = ") (not #t)

(display " not F = ") (not #f) (newline)

;;-----------

(display "Tabela de valores ->") (newline)

(define (-> p q) (or (not p) q))

(display " T -> T = ") (-> #t #t)

(display " T -> F = ") (-> #t #f)

(display " F -> T = ") (-> #f #t)

(display " F -> F = ") (-> #f #f) (newline)

;;-----------

(display "Tabela de valores <->") (newline)

(define (<-> p q) (and (-> p q) (-> q p)))

(display " T <-> T = ") (<-> #t #t)

(display " T <-> F = ") (<-> #t #f)

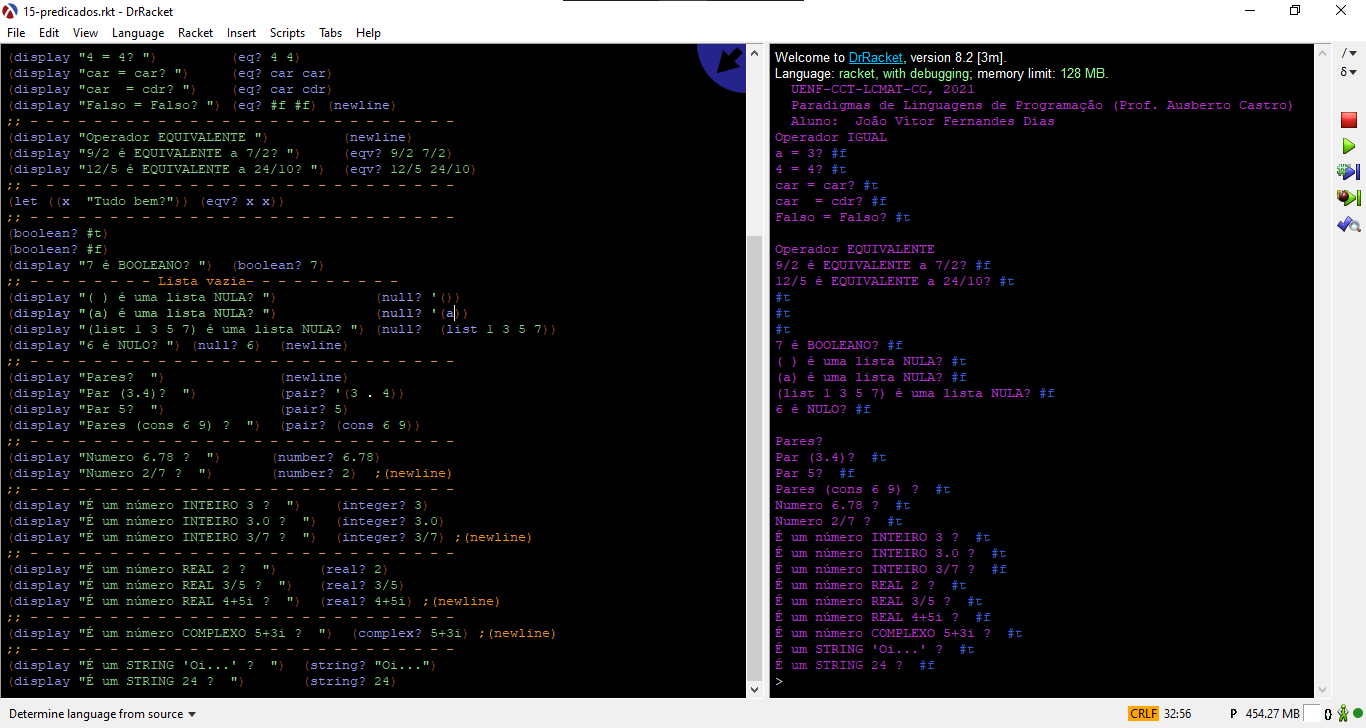
(display " F <-> T = ") (<-> #f #t)

(display " F <-> F = ") (<-> #f #f)

#### Arquivo 15-predicados.rtk 21h28 – 21h56 DÚVIDA

##### Executar o programa e indicar o seu conteúdo 21h28 – 21h50 DÚVIDA

###### Print



###### Explicação

Este código exemplifica o uso de predicados (?) (não sei se é assim que se deve chamar).

DÚVIDA: O que são de fato os predicados? Expressões com checagem?

Os predicados são expressões de checagem (são mesmo?) que averiguam certa propriedade de um determinado valor. Neste código são mostrados alguns, com alguns exemplos de cada um. Os predicados mostrados foram:

eq? Confere se os dois valores comparados são iguais

eqv? Confere se os dois valores comparados são equivalentes

boolean? Confere se o valor comparado é booleano

null? Confere se o valor/lista comparado(a) é nulo(a)

pair? Confere se o valor/lista comparado(a) é um par

number? Confere se o valor comparado é um número

integer? Confere se o valor comparado é um número inteiro

real? Confere se o valor comparado é um número real

complex? Confere se o valor comparado é um número complexo

string? Confere se o valor comparado é uma string

##### Testar os predicados: 21h50 – 21h56

###### Predicados

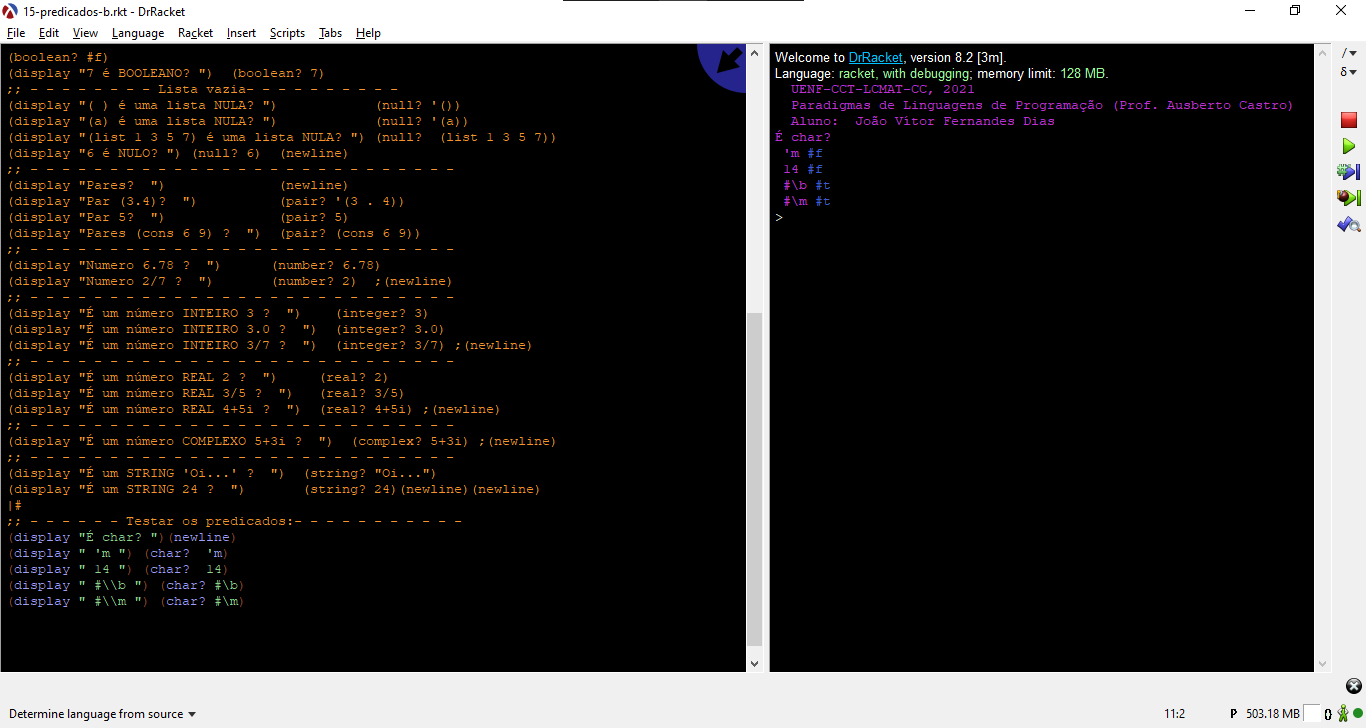
(char? ´m)

(char? 14)

(char? #\b)

(char? #\m)

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline)

#|

;; ---------------------------------------------------

;; Predicados

;; ---------------------------------------------------

(display "Operador IGUAL ") (newline)

(display "a = 3? ") (eq? 'a 3)

(display "4 = 4? ") (eq? 4 4)

(display "car = car? ") (eq? car car)

(display "car = cdr? ") (eq? car cdr)

(display "Falso = Falso? ") (eq? #f #f) (newline)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "Operador EQUIVALENTE ") (newline)

(display "9/2 é EQUIVALENTE a 7/2? ") (eqv? 9/2 7/2)

(display "12/5 é EQUIVALENTE a 24/10? ") (eqv? 12/5 24/10)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(let ((x "Tudo bem?")) (eqv? x x))

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(boolean? #t)

(boolean? #f)

(display "7 é BOOLEANO? ") (boolean? 7)

;; - - - - - - - - Lista vazia- - - - - - - - - -

(display "( ) é uma lista NULA? ") (null? '())

(display "(a) é uma lista NULA? ") (null? '(a))

(display "(list 1 3 5 7) é uma lista NULA? ") (null? (list 1 3 5 7))

(display "6 é NULO? ") (null? 6) (newline)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "Pares? ") (newline)

(display "Par (3.4)? ") (pair? '(3 . 4))

(display "Par 5? ") (pair? 5)

(display "Pares (cons 6 9) ? ") (pair? (cons 6 9))

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "Numero 6.78 ? ") (number? 6.78)

(display "Numero 2/7 ? ") (number? 2) ;(newline)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "É um número INTEIRO 3 ? ") (integer? 3)

(display "É um número INTEIRO 3.0 ? ") (integer? 3.0)

(display "É um número INTEIRO 3/7 ? ") (integer? 3/7) ;(newline)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "É um número REAL 2 ? ") (real? 2)

(display "É um número REAL 3/5 ? ") (real? 3/5)

(display "É um número REAL 4+5i ? ") (real? 4+5i) ;(newline)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "É um número COMPLEXO 5+3i ? ") (complex? 5+3i) ;(newline)

;; - - - - - - - - - - - - - - - - - - - - - - - - - - -

(display "É um STRING 'Oi...' ? ") (string? "Oi...")

(display "É um STRING 24 ? ") (string? 24)(newline)(newline)

|#

;; - - - - - - Testar os predicados:- - - - - - - - - - -

(display "É char? ")(newline)

(display " 'm ") (char? 'm)

(display " 14 ") (char? 14)

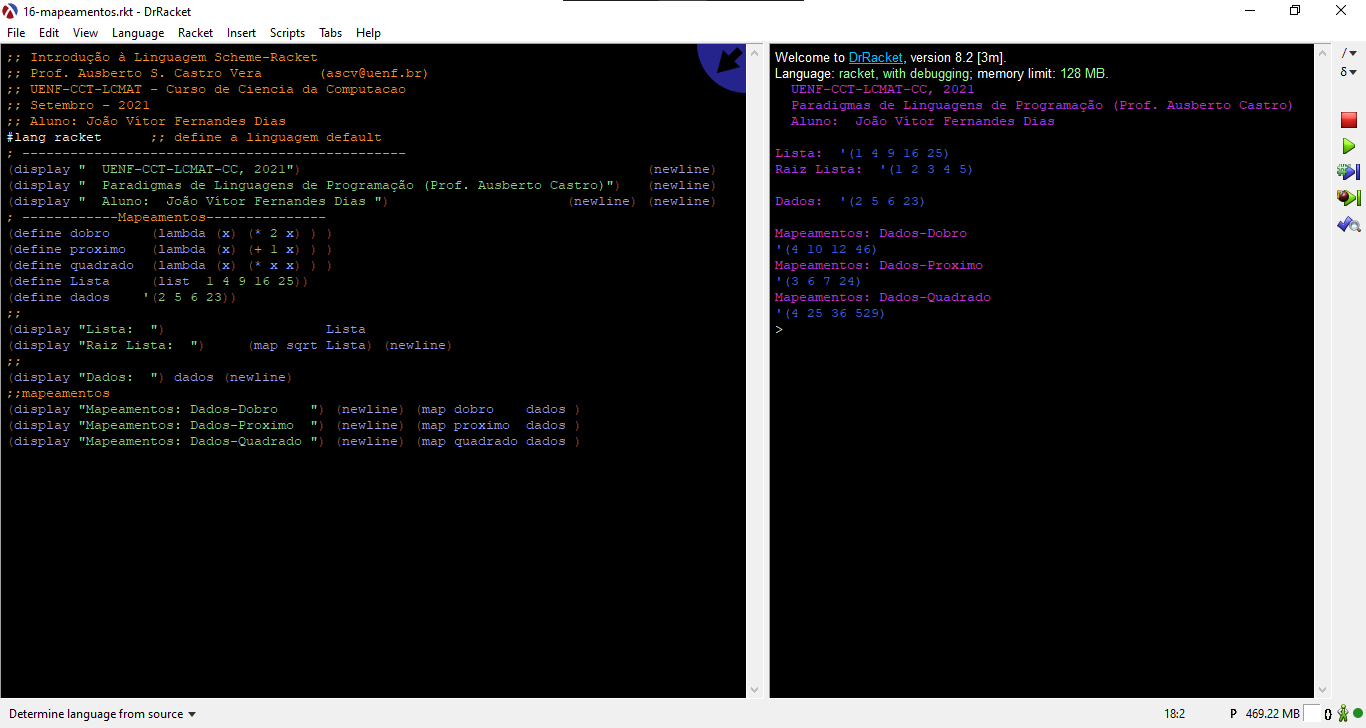
(display " #\\b ") (char? #\b)

(display " #\\m ") (char? #\m)

#### Arquivo 16-mapeamentos.rtk 08/09/21 09h06 – 09h35 DÚVIDA

##### Executar o programa e indicar o que ele faz 09h06 – 09h29 DÚVIDA

###### Print



###### Explicação

Este código ilustra o uso da função (?) map.

**DÚVIDA: “map” é uma função?**

A função (?) map permite que um procedimento seja efetuado por sobre toda uma lista de dados. Para isso, nesse código foram definidas 3 funções: dobro, próximo e quadrado.

Dobro recebe um número e retorna o seu valor multiplicado por dois

Próximo recebe um número e retorna o seu valor somado a um

Quadrado recebe um número e retorna o seu valor multiplicado a ele mesmo

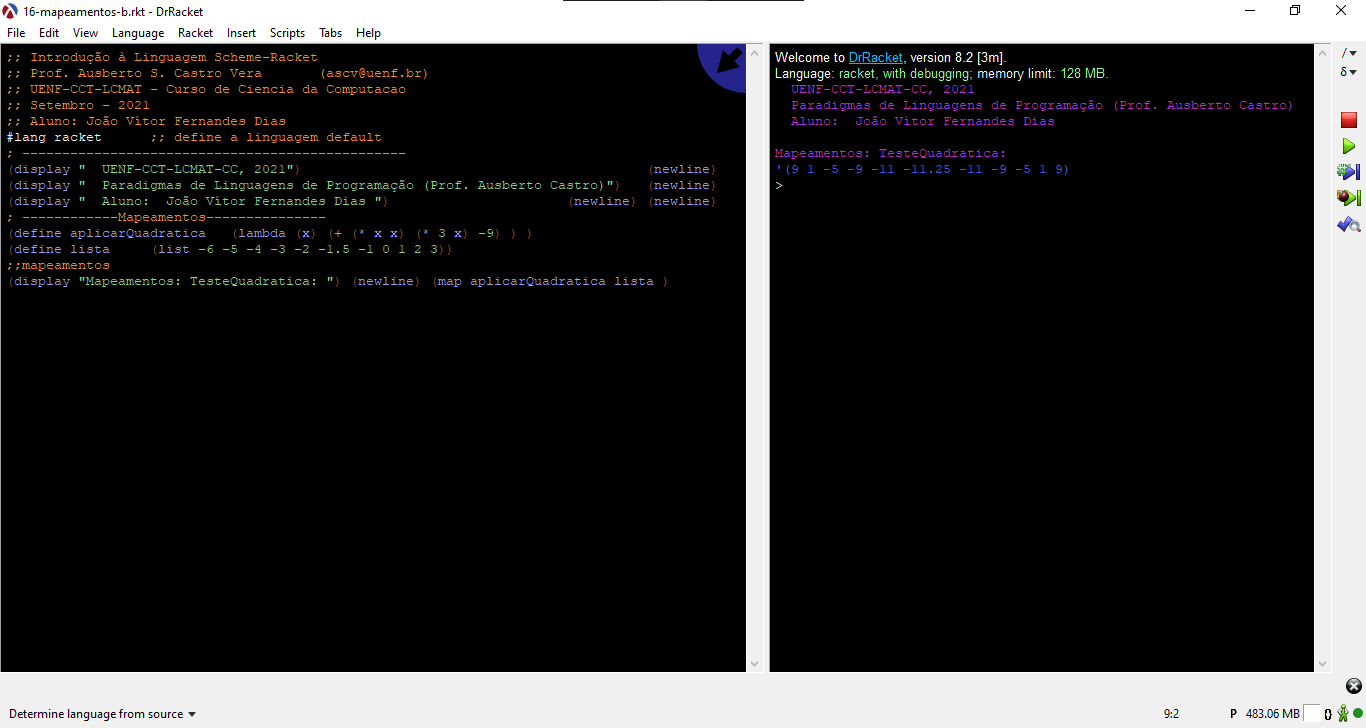
Não são definidas duas listas sobre as quais será utilizado as funções criadas em conjunto com a função map.

##### Construir um NOVO programa que faça o seguinte mapeamento 09h29 – 09h35

###### Mapeamento

x ⎯→ x2 + 3x - 9

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

#lang racket ;; define a linguagem default

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

; ------------Mapeamentos---------------

(define aplicarQuadratica (lambda (x) (+ (\* x x) (\* 3 x) -9) ) )

(define lista (list -6 -5 -4 -3 -2 -1.5 -1 0 1 2 3))

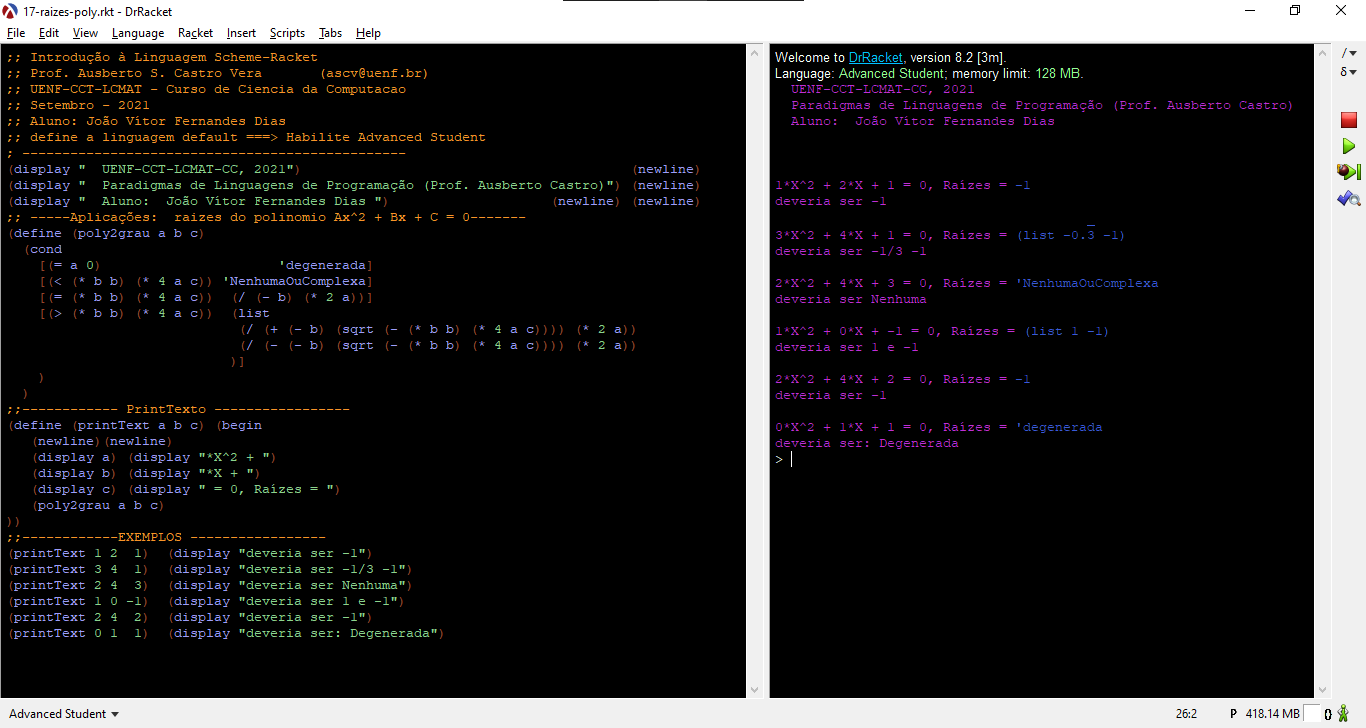
;;mapeamentos

(display "Mapeamentos: TesteQuadratica: ") (newline) (map aplicarQuadratica lista )

#### Arquivo 17-raizes-poly.rtk Aplicações: Raízes de polinômios 09h35 – 10h07

##### Executar o programa e explicar o conteúdo e os resultados 09h35 – 09h57

###### Print



###### Explicação

Este código ilustra o uso de polinômios. Com a função “polygrau2” que recebe como parâmetro os coeficientes A, B e C de uma equação de segundo grau, ele calcula que tipo de polinômio ele é:

Caso o seu coeficiente A seja igual a zero, ela é degenerada, pois não é uma função de segundo grau.

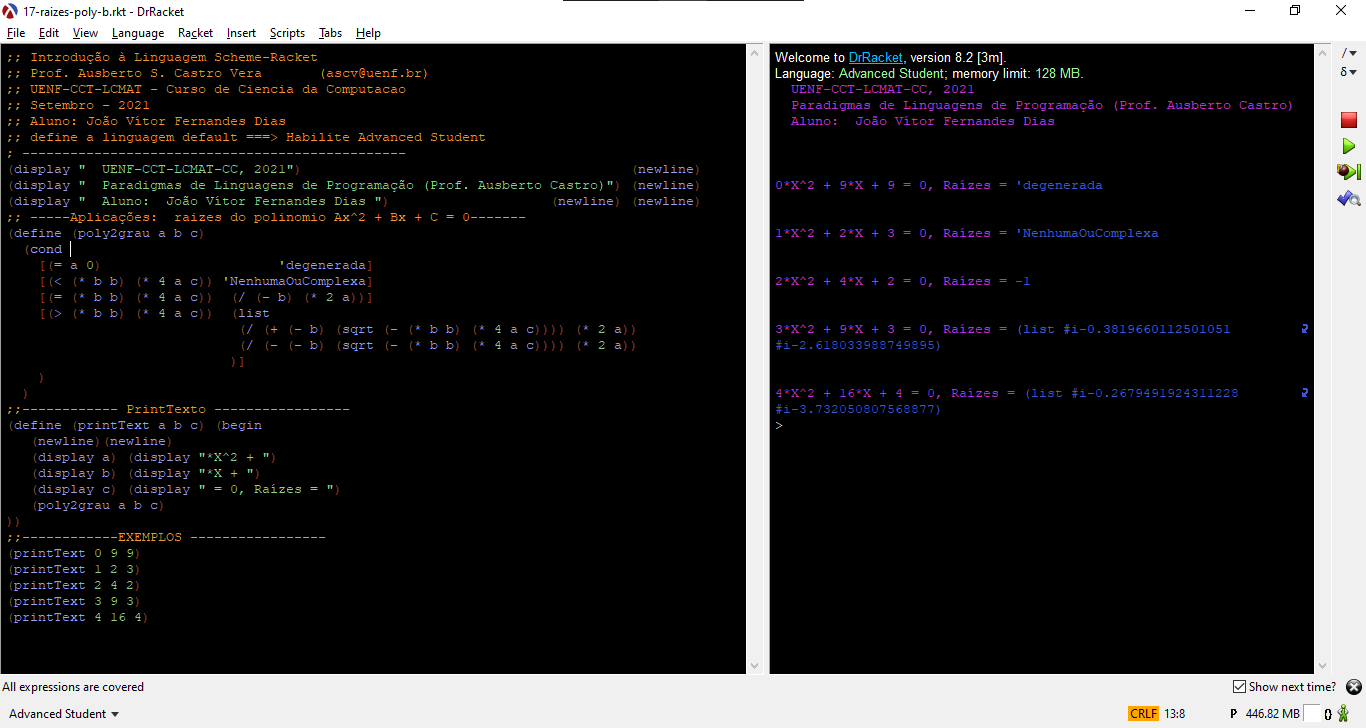
Caso B² seja menor que 4\*A\*C, não haverá raiz real.

Caso B² seja maior que 4\*A\*C, cria uma lista composta pela solução de X1 e X2 utilizando a fórmula de Bhaskara.

(Abaixo eu fiz um procedimento que imprime de forma mais compacta o polinômio com seus coeficientes e também chama a função poly2grau.)

##### Fazer testes para outros cinco polinômios de segundo grau 09h59 – 10h07

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

;; define a linguagem default ===> Habilite Advanced Student

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

;; -----Aplicações: raizes do polinomio Ax^2 + Bx + C = 0-------

(define (poly2grau a b c)

(cond

[(= a 0) 'degenerada]

[(< (\* b b) (\* 4 a c)) 'NenhumaOuComplexa]

[(= (\* b b) (\* 4 a c)) (/ (- b) (\* 2 a))]

[(> (\* b b) (\* 4 a c)) (list

(/ (+ (- b) (sqrt (- (\* b b) (\* 4 a c)))) (\* 2 a))

(/ (- (- b) (sqrt (- (\* b b) (\* 4 a c)))) (\* 2 a))

)]

)

)

;;------------ PrintTexto -----------------

(define (printText a b c) (begin

(newline)(newline)

(display a) (display "\*X^2 + ")

(display b) (display "\*X + ")

(display c) (display " = 0, Raízes = ")

(poly2grau a b c)

))

;;------------EXEMPLOS -----------------

(printText 0 9 9)

(printText 1 2 3)

(printText 2 4 2)

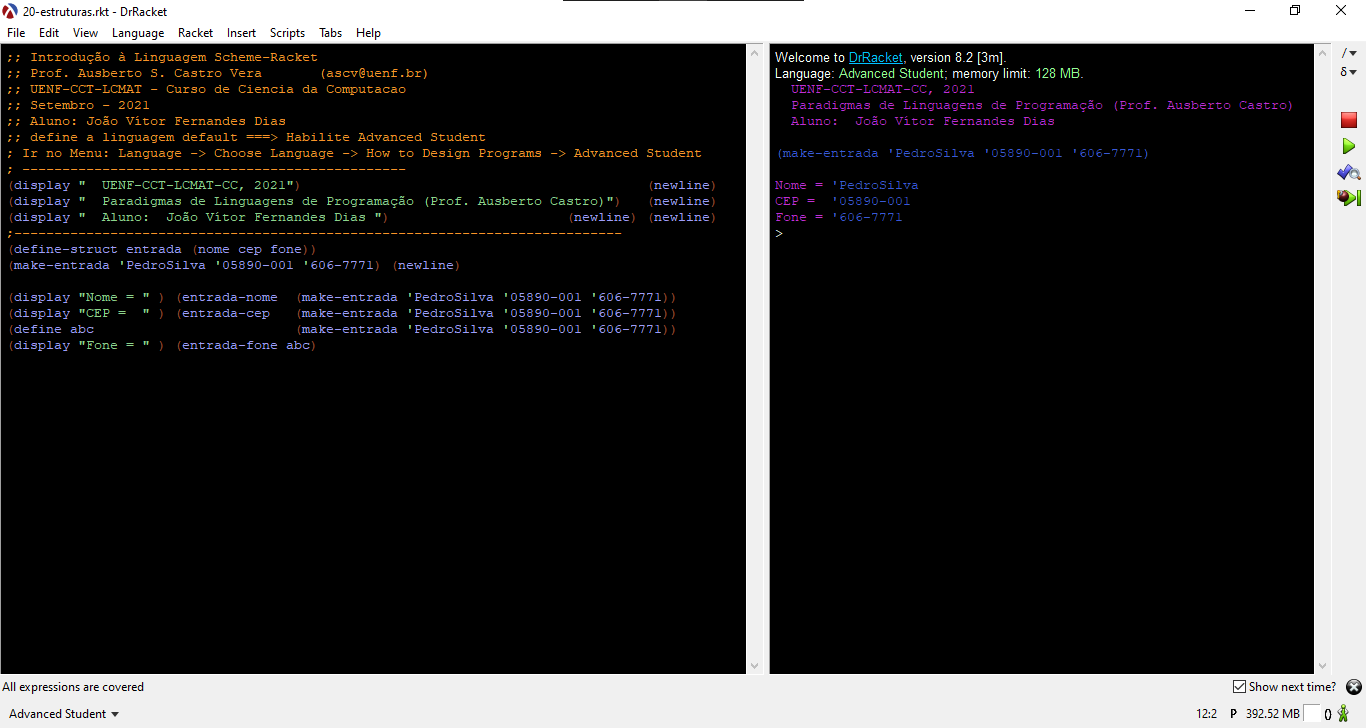
(printText 3 9 3)

(printText 4 16 4)

#### Arquivo 20-estruturas.rtk Aplicações: Estruturas de dados 10h07 – 10h59

##### Executar o programa e explicar o conteúdo e os resultados 10h07 – 10h32

###### Print



###### Explicação

Esse código ilustra o uso de estruturas

A estrutura “entrada” é definida com a função “define-struct” e possuirá 3 campos: nome, cep e fone.

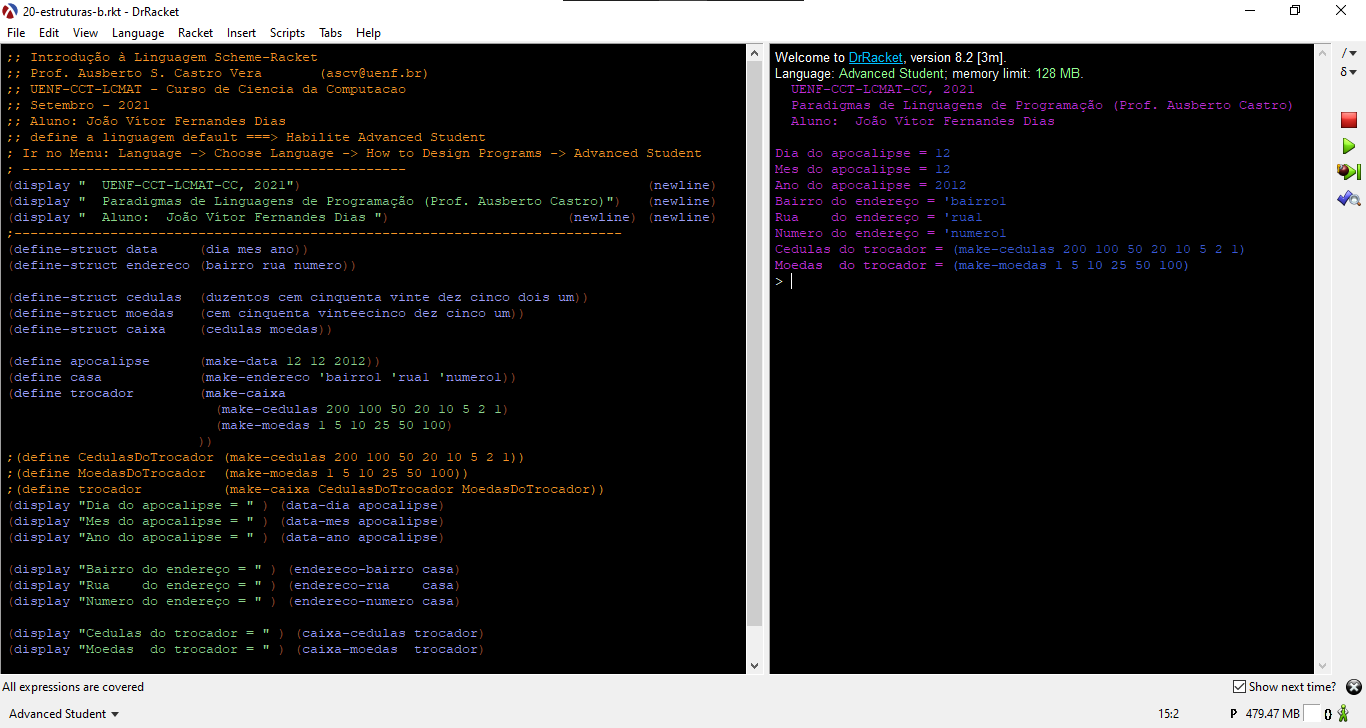
Com a função “make-NomeDaEstrutura” é possível criar uma estrutura

Para exibir apenas um dos campos da estrutura, é necessário utilizar da seguinte forma: “NomeDaEstrutura-NomeDoCampo” seguido da estrutura criada ou de uma função que retorne uma estrutura criada.

E assim é feito utilizando os termos “entrada-nome”, “entrada-cep”, “entrada-fome”.

##### Em um novo programa defina pelo menos outras TRÊS estruturas diferentes 10h34 – 10h59

###### Print



###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

;; define a linguagem default ===> Habilite Advanced Student

; Ir no Menu: Language -> Choose Language -> How to Design Programs -> Advanced Student

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline)

;----------------------------------------------------------------------------

(define-struct data (dia mes ano))

(define-struct endereco (bairro rua numero))

(define-struct cedulas (duzentos cem cinquenta vinte dez cinco dois um))

(define-struct moedas (cem cinquenta vinteecinco dez cinco um))

(define-struct caixa (cedulas moedas))

(define apocalipse (make-data 12 12 2012))

(define casa (make-endereco 'bairro1 'rua1 'numero1))

(define trocador (make-caixa

(make-cedulas 200 100 50 20 10 5 2 1)

(make-moedas 1 5 10 25 50 100)

))

;(define CedulasDoTrocador (make-cedulas 200 100 50 20 10 5 2 1))

;(define MoedasDoTrocador (make-moedas 1 5 10 25 50 100))

;(define trocador (make-caixa CedulasDoTrocador MoedasDoTrocador))

(display "Dia do apocalipse = " ) (data-dia apocalipse)

(display "Mes do apocalipse = " ) (data-mes apocalipse)

(display "Ano do apocalipse = " ) (data-ano apocalipse)

(display "Bairro do endereço = " ) (endereco-bairro casa)

(display "Rua do endereço = " ) (endereco-rua casa)

(display "Numero do endereço = " ) (endereco-numero casa)

(display "Cedulas do trocador = " ) (caixa-cedulas trocador)

(display "Moedas do trocador = " ) (caixa-moedas trocador)

#### Arquivo 30-entrada.rtk 11h09 – 12H41 **DÚVIDA**

##### Executar o programa e explicar o conteúdo e os resultados 11h09 – 11h35

###### Print



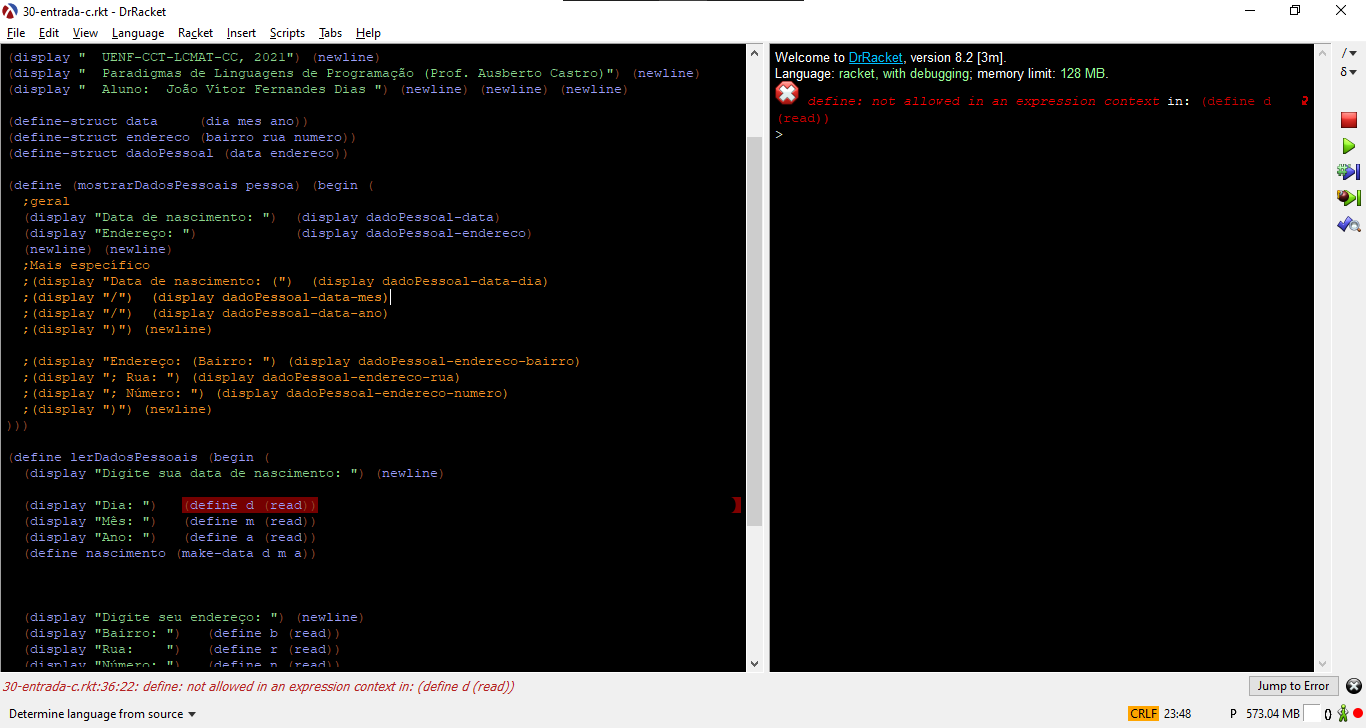
###### Explicação

Esse código exemplifica o uso da entrada de dados pelo usuário. Utiliza a função read para permitir tal feito.

Esse código define a função de média, em seguida define os valores das notas A, B e C de acordo com o input do usuário, calcula a média com a função previamente definida, mostra seu valor e então confere se essa média é maior do que 5, se for, “Aprovado” será mostrado na tela, senão, “Reprovado” será mostrado.

##### Criar um programa NOVO que faça a leitura de dados pessoais de duas pessoas (utilize entrada de dados) **DÚVIDA** 11h35 – 12h41

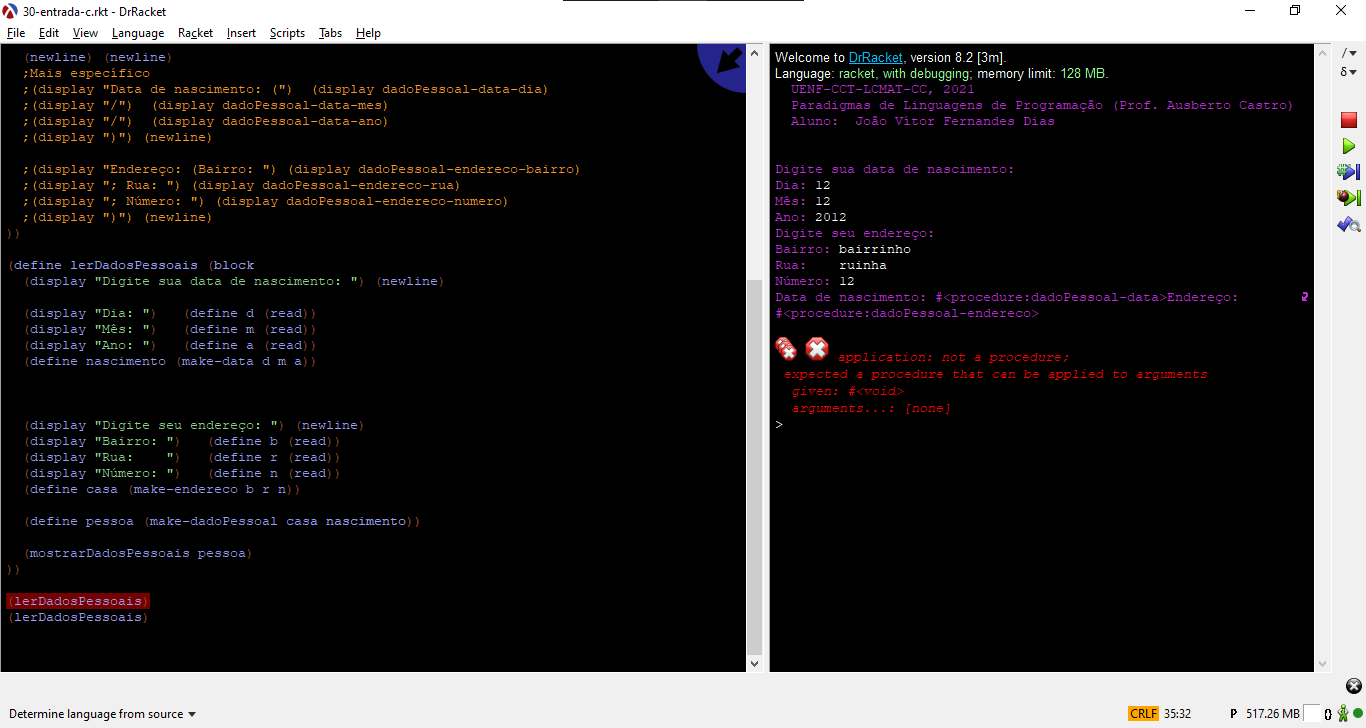
###### Print



DÚVIDA: Por que o define não está funcionando?

Utilizei a biblioteca Block e removi uns parênteses

Mas...



DÚVIDA: Qual é o problema agora?

###### Código

;; Introdução à Linguagem Scheme-Racket

;; Prof. Ausberto S. Castro Vera (ascv@uenf.br)

;; UENF-CCT-LCMAT - Curso de Ciencia da Computacao

;; Setembro - 2021

;; Aluno: João Vítor Fernandes Dias

#lang racket ;; define a linguagem default

(require racket/block)

; ------------------------------------------------

(display " UENF-CCT-LCMAT-CC, 2021") (newline)

(display " Paradigmas de Linguagens de Programação (Prof. Ausberto Castro)") (newline)

(display " Aluno: João Vítor Fernandes Dias ") (newline) (newline) (newline)

(define-struct data (dia mes ano))

(define-struct endereco (bairro rua numero))

(define-struct dadoPessoal (data endereco))

(define (mostrarDadosPessoais pessoa) (block

;geral

(display "Data de nascimento: ") (display dadoPessoal-data)

(display "Endereço: ") (display dadoPessoal-endereco)

(newline) (newline)

;Mais específico

;(display "Data de nascimento: (") (display dadoPessoal-data-dia)

;(display "/") (display dadoPessoal-data-mes)

;(display "/") (display dadoPessoal-data-ano)

;(display ")") (newline)

;(display "Endereço: (Bairro: ") (display dadoPessoal-endereco-bairro)

;(display "; Rua: ") (display dadoPessoal-endereco-rua)

;(display "; Número: ") (display dadoPessoal-endereco-numero)

;(display ")") (newline)

))

(define lerDadosPessoais (block

(display "Digite sua data de nascimento: ") (newline)

(display "Dia: ") (define d (read))

(display "Mês: ") (define m (read))

(display "Ano: ") (define a (read))

(define nascimento (make-data d m a))

(display "Digite seu endereço: ") (newline)

(display "Bairro: ") (define b (read))

(display "Rua: ") (define r (read))

(display "Número: ") (define n (read))

(define casa (make-endereco b r n))

(define pessoa (make-dadoPessoal casa nascimento))

(mostrarDadosPessoais pessoa)

))

(lerDadosPessoais)

(lerDadosPessoais)