Linguagem Arcadia

Marcelo Norberto
marcelonorberto07@gmail.com

A linguagem Arcadia baseia-se em alguns conceitos da linguagem C, Ela é pode implementar diversos programas.

Variáveis

O programa tem apenas um tipo de variável que aceita valores decimais como 1.0,0.5
Embora para operações com números decimais ocorra alguns erros de precisão .
Declaração: para declarar uma variável deve se iniciar com '**vars**' seguido do nome que a variável terá.Ex:***vars** a;*
Atribuição: para atribuir um valor a uma variável deve se digitar o nome da variável seguido de '**=**' e do valor desejado.Ex: ***a = 3;***
Pode se declarar e atribuir na mesma linha.Ex:***vars a = 3;***
Expressões: pode se utilizar expressões de soma,subtração,divisão e multiplicação apenas com dois operadores.Ex:***a = 3 *2;***
Pode se declarar e atribuir uma expressão na mesma linha.Ex:***vars a = 3*2;***
Comparação:para utilizar uma comparação deve usar um função de laço ou de controle de fluxo.Ex:***loop(a<5); select(a==4);***

Funções

loop(**comparação**): Função de laço, ela é executada caso a comparação seja verdadeira.
Ex:***loop(a!=5){}***
select(**comparação**)**:** Função usada para controlar o fluxo do programa, ela é executada caso a comparação seja verdadeira.Ex:***select(a==7){}***
screen(**String s=variável**): Função usada para mostrar mensagens na tela.
Ex: ***screen(oi= a);***
screin(**variável=String s**): Função usada para mostrar uma caixa de dialogo na qual se pede que o usuário digite um valor.
Ex: ***screin(a =digite um valor);***

Classe e Protótipos dos Métodos

Classe Arcadia:Esta é a main do programa apartir dela todas as demais classes são executadas, bem como o onde arquivo contendo código fonte é analisado e seu código é executado.

Classe Agiliza: Esta classe reduz a repetição de códigos, além de separar uma String em partes menores e testa se um nome de variável é valido.
public static int percore(String[] texto,int nlinhas)
Este método percorre as linhas do código apartir de um método loop ou select, retorna o numero de linhas ate onde se estende o método, retorna -1 caso o método não tenha o caractere de terminação.
public static String crescente(String s,int i,int aux)
Este método retorna uma String apenas com caracteres alfanuméricos no intervalo da String passada como parâmetro apartir do valor de i ate o valor de aux. Retorna null se o espaço da String definido pelos parâmetros for nula ou a String for nula.

public static String testanum(String s)
Este método testa se na String passada há apenas números, Retorna a String caso haja apenas números ou retorna null caso haja um caractere diferente de um numero.
public static String testanome(String s)
Este método testa se o primeiro caractere da String é uma letra, Retorna null caso o primeiro caractere for um numero ou caractere especial caso contrario retorna a própria String.
public static String decrescente(String s,int i,int aux)
Este método retorna uma String apenas com caracteres alfanuméricos no intervalo da String passada como parâmetro apartir do valor de aux ate o valor de i. Retorna null se o espaço da String definido pelos parâmetros for nula ou a String for nula.

Classe Analise: Esta classe elimina espaços antes do primeiro caractere alfanumérico escrito na String linha, também testa se a linha do código tem ou não o caractere de terminação.
public static Variavelimite[] frase(String linha,Variavelimite[] lim)
Este método percorre uma linha do código ignorando espaços antes do primeiro caractere alfanumérico ela para quando encontra o caractere de terminação ou chega ao fim da String. Retorna um vetor da Variavelimite com duas posições contendo o numero de inicio e fim da linha de acordo com os critérios acima sendo que a primeira posição do vetor deve ter um valor menor que o da segunda, caso contrario significa que a String não tem caractere de terminação.

Classe Calculos: Esta classe implementa métodos matemáticos
public static double soma(double a,double b)
public static double subt(double a,double b)
public static double divi(double a,double b)
public static double mult(double a,double b)
public static double mod(double a,double b)
Estes métodos retornam o valor de uma operação matemática sendo na ordem soma,subtração,divisão , multiplicação e o resto da divisão.
public static boolean diferente(double a,double b)
public static boolean igual(double a,double b)
public static boolean menorigual(double a,double b)
public static boolean maiorigual(double a,double b)
public static boolean menor(double a,double b)
public static boolean maior(double a,double b)
Estes métodos retornam true caso a comparação seja verdadeira e caso contrario retorna false.

Classe Compare: Esta classe executa o método de comparação,utiliza em laços e controladores de fluxo.
public static int compara(String linha,Executa menu,Variavel[] seq)
Este método retorna 1 caso a comparação seja verdadeira, 0 caso seja falsa, e -1 caso haja algum erro na expressão de comparação.

Classe Converte: Esta classe converte um arquivo com o código em um vetor de Strings usada para a implementação das demais classes.
public static String[] convertstring(String[] args,Variavelimite[] lim)
Este método retorna um vetor de Strings apartir de um arquivo com o código fonte. Retorna null caso o arquivo não exista ou em uma linha especifica não haja o caractere de terminação adequado.

Classe Declara: Esta classe manipula a variável do programa,inserindo e atribuindo valores a novas variáveis.

public static Variavel getVariavel(String nome,Variavel[] seq)

Este método retorna uma variável se no vetor de Variavel houver uma com o mesmo nome passado na String.Retorna null caso contrario.

public static void setVariavel(Variavel v,Variavel[] seq)

Este método insere uma nova variável ao vetor de variáveis.

public static Variavel[] insere(String s,Variavel[] seq,String linha)

Este método verifica se a String s contem um nome valido para uma variável e se houver verifica se ela já existe retorna o vetor de variáveis sem modificações, caso não exista insere ela ao vetor de variáveis,se o nome for invalido retorna null.

public static Variavel[] atribui(String s,Executa menu,Variavel[] seq)

Este método verifica se a String s apartir do inicio ate um valor definido na variável menu, contem um nome valido para uma variável e se houver verifica se ela já existe então verifica se o valor contido na String s apartir do valor definido na variável menu ate o fim da String s é uma atribuição valida e se for então atribui a variável e retorna o vetor de variáveis. Retorna null caso haja algum erro na String s.

Classe Executa: Esta classe é usada para o programa principal saber quais métodos serão usados para interpretar cada linha do código fonte.

private String nome;

public boolean status;

public int indice;

public Executa()

Construtor da classe seta o valor da variável status como false.

public void setStatus()

Modifica o valor da variável status.

public void setNome(String s)

Este método faz com que o atributo nome recebe a String s.

public String getNome()

Retorna a String nome da variável.

Classe Expressao: Esta classe analise linhas do código fonte que contenham expressões matemáticas

public static String opera(String s,Executa[] menu,Variavel[] seq)

Este método verifica se a String s contem números ou uma ou mais variáveis e se houver verifica se elas existem, caso não existam retorna null, também retorna null caso haja um expressão ilegal ou ilegível pelo programa. Retorna o resultado dos números ou variáveis após executar a operação matemática adequada.

Classe Funcao: Esta classe implementa métodos de laço e seleção de fluxo.

public static Variavel[] repeticao(int i,int f,String[] texto,Variavel[] vars,Executa inicio)

Este método executa a função de laço no programa. Caso não ocorra erros retorna o vetor vars, se houver erros retorna null.

public static Variavel[] selecao(int i,int f,String[] texto,Variavel[] vars)

Este método executa a função de seleção de fluxo no programa. Caso não ocorra erros retorna o vetor vars, se houver erros retorna null.

Classe Interpretador: Esta classe interpreta cada linha do código fonte,seta valores na respectiva variável menu

public static Executa[] interpreta(String linha,Executa[] menu)

Este método seta os atributos status e índice do vetor menu de acordo com o código escrito na linha analisada.

Classe Tela: Esta classe implementa métodos de entrada e saída.

public static Variavel[] dialogbox(String linha,Executa separa,Variavel[] seq)

Este método permite mostra uma caixa de dialogo mostrando uma mensagem ao usuário sendo que se pode solicitar uma entrada do usuário. Retorna null caso haja um erro no código executado, caso o código execute sem erros retorna o vetor seq.

Classe Variavel: Esta classe implementa a Variavel principal do programa.

private String nome;
public double valor;

public void setNome(String s)

Este método faz com que o atributo nome recebe o valor da String s.

public void setValor(double n)

Este método faz com que o atributo valor recebe o valor de n.

public String getNome()

Este método retorna o valor do atributo nome.

public double getValor()

Este método retorna o valor do atributo valor.

Classe Variavelimite: Esta classe implementa a Variavel responsável por reduzir uma linha do código fonte ignorando espaços antes do primeiro caractere alfanumérico e comentários após o caractere de terminação

private String nome;
public int valor;

public void setNome(String s)

Este método faz com que o atributo nome receba o valor da String s.

public void setValor(int n)

Este método faz com que o atributo valor recebe o valor de n.

public String getNome()

Este método retorna o valor do atributo nome.

public int getValor()

Este método retorna o valor do atributo valor.

Exemplos de Código na Linguagem Arcadia

Declarações e atribuições

```
1º
vars a;
a = 2;
screen(“a é “, a);
2º
vars b = 3;
screen(“b é “, b);
3º
vars a = 2.3;
vars b;
```

Expressões

```
vars a = 2.3;
vars b;
b = a + 1;
vars c = b*2;
b = b - 1;
 c = c - 1;
a = a + 2 - 1;
```

Laço

```
vars a = 2.3;
vars b;
b = a + 1;
vars c = b*2;
loop(c>2){
	c = c - 1;
	screen(“”, c);
}
```

Controle de Fluxo

```
vars a = 2.3;
vars b;
b = a + 1;
select(a < b){
		vars c = b*2;
		a = a+1;
}
```

Aninhamento

```
vars a = 2.3;
vars b;
b = a + 1;
select(a < b){
		vars c = b*2;
		loop(c>2){
				b = 7;
				select(b<3){
				b = b - 1;
				}
				c = c - 1;
		}
		a = a+1;
}
a = a + 2 - 1;
```