Universidade Federal de Alagoas
Instituto de Computação
Ciência da computação

Compiladores - Especificação

Harlan

José Leandro da Silva Filho

Maceió, 15 de Março de 2018

Conteúdo

Conteúdo 1

1. Introdução 2

2. Estrutura 2

3. Tipos 2

4. Operadores 2

5. Identificadores 2

6. Comentários 3

7. Instruções 3

7.1 if - else 3

7.2 while 3

7.3 for 3

7.4 input() 4

7.5 print() 4

8. Atribuição 4

9. Funções 4

10. Escopo 5

11. Exemplos de código 5

12. Especificação dos tokens 7

1. Introdução

Harlan é uma linguagem de programação estaticamente tipada, imperativa e fortemente influenciada por C e Python.

2. Estrutura

Funções e variáveis podem ser declaradas em qualquer lugar do programa e são acessíveis a partir do momento em que forem declaradas. Variáveis quando declaradas só são acessíveis pelo escopo local. A execução de todo programa é iniciada a partir da função main() que deve ser obrigatoriamente definida sem parâmetros.

3. Tipos

Harlan possui o seguinte conjunto de tipos: inteiro (int), ponto flutuante (float), cadeia de caractere (string), booleano (bool) e arrays unidimensionais (definidos usando o operador “[]”). Não existe coerção de tipo em Harlan. Constantes numéricas de ponto flutuante são definidas por digitos com decimais separados por “.”.

4. Operadores

A tabela abaixo descreve a precedência de operadores em Harlan, da menor precedência para a maior.

or	“Ou” booleano
and	“E” booleano
not	Negação
<, <=, >, >=, !=, ==, in	Comparação
+, -	Adição/Concatenação e Subtração
*, /, %	Multiplicação, Divisão e Resto
-x, +x	Unário negativo e positivo

A avaliação de operadores que possuírem a mesma precedência segue a regra de associatividade da esquerda para a direita.

5. Identificadores

Identificadores são obrigatoriamente iniciados por letras. Subsequentemente ao primeiro caractere, podem conter letras e digitos. Identificadores não podem conter espaços em branco.

6. Comentários

Comentários são feitos usando o operador “//”.

7. Instruções

7.1 if - else

A instrução “if” é usada da seguinte forma:

```
if(condicao) {
        corpo_do_if
}
```

A cláusula “else” pode ser opcionalmente adicionada a instrução para definir um corpo de instruções alternativo para ser executado caso a “condicao” não seja satisfeita.

```
if(condicao) {
        corpo_do_if
} else {
        corpo_do_else
}
```

No caso de instruções if aninhadas, a cláusula else pertencerá a última instrução if fechada.

7.2 while

A instrução while executa seu bloco até que a condição (expressão booleana) estabelecida não seja mais verdadeira.

```
while(condicao) {
        corpo_do_while
}
```

7.3 for

A instrução for é usada para criar um loop que será executado uma quantidade predeterminada de vezes.

```
for i in range(start, stop, step) {
        corpo_do_for
}
```

A função embutida range() retorna um iterador que será percorrido pela instrução durante sua execução. O parametro step indica o valor a ser incrementado em cada iteração.

7.4 input()

Entrada de dados em Harlan é obtida através do retorno da função embutida input().

7.5 print()

A função print(format, ...) exibirá o conteúdo dos parâmetros subsequentes ao primeiro na saída padrão em um formato definido pelo primeiro parâmetro (format).
Exemplo:

```
print("%s", "Hello, world!");
```

8. Atribuição

Em Harlan, atribuição é feita através do operador de atribuição “=”.
Exemplo:

```
int x = 15;
int y = 10 + x;
```

9. Funções

Funções em Harlan são definidas da seguinte forma:

```
int foo(int x) {
        return x*x;
}
```

O tipo dos parâmetros da função são especificados antes dos identificadores de cada parametro. O retorno da função é definido antes do identificador da função.

10. Escopo

Escopos são delimitador por chaves (“{}”).

11. Exemplos de código

Abaixo alguns exemplos de códigos escritos em Harlan. Disponíveis também em https://github.com/jleandrof/harlan-compiler/.

Olá mundo:

```
main() {
        print("Olá mundo!");
}
```

Série fibonacci:

```
fib(int n) {
    int a = 0;
    int b = 1;
    int next;

    print("%d, %d", a, b);
    next = a + b;

    while(next <= n) {
        print(", %d", next);
        a = b;
        b = next;
       next = a + b;
    }
}

main() {
    fib(input());
}
```

Shellsort:

```
int maxsize = 10;

int[] shellsort(int[] array) {
    int inner, outer;
    int valueToInsert;
    int interval = 1;
    int elements = maxsize;
    int i = 0;

    while(interval  <= elements / 3) {
        interval = interval * 3 + 1;
```

```
    }

    while(interval > 0) {
        outer = interval;
       for outer in range(outer, elements) {
           valueToInsert = array[outer];
           inner = outer

               while(inner  >  (interval  -  1)  and  array[inner  -  interval]  >=
valueToInsert) {
               array[inner] = array[inner - interval];
               inner = inner - interval;
           }

           array[inner] = valueToInsert;
       }

       interval = (interval - 1) / 3;
       i = i + 1;
    }

    return array;
}

print_array(int[] array) {

    int i;

    print("[");
    for i in range(0, maxsize) {
       print("%d ", array[i]);
    }
    print("]\n");
}

main() {
    int array = [5, 7, 2, 0, 9, 6, 1, 3, 4, 8];

    print("Array Original: ");
    print_array(array);

    print("Array Ordenado: ");
    print_array(shellsort(array));
}
```

12. Especificação dos tokens

Expressões regulares

```
KWMAIN = '(main)(?!\w)'
KWINPUT = '(input)(?!\w)'
KWPRINT = '(print)(?!\w)'
KWIF = '(if)(?!\w)'
KWELSE = '(else)(?!\w)'
KWFOR = '(for)(?!\w)'
KWWHILE = '(while)(?!\w)'
KWRANGE = '(range)(?!\w)'
STRTYPE = '(string)(?!\w)'
ITYPE = '(int)(?!\w)'
FTYPE = '(float)(?!\w)'
BTYPE = '(bool)(?!\w)'
CTEFLOAT = '[+-]?\d+\.\d+(?!\w)'
CTEINT = '[+-]?\d+(?!\w)'),
LTBOOL = '(true|false)(?!\w)'
OPREL = '>=|>|<=|<'
OPEQ = '==|!='
OPAT = '='
OPMBR = '(in)(?!\w)'
OPAD = '\+|-'
OPML = '\*|/|%'
OPCONJ = '(and)(?!\w)'
OPDISJ = '(or)(?!\w)'
OPNEG = '(not)(?!\w)'
ST = ';'
CLN = ','
PARST = '\('
SQBRST = '\['
CLBRST = '\{'
PAREND = '\)'
SQBREND = '\]'
CLBREND = '\}'
LTSTRING = '\".*?\"'
KWRETURN = '(return)(?!\w)'
ID = '[a-zA-Z_]+[0-9]*[a-zA-Z_]*'
```

Categorias

[Token]: [Categoria]

'KWMAIN': 1
'KWINPUT': 2
'KWPRINT': 3
'KWIF': 4
'KWELSE': 5
'KWFOR': 6
'KWWHILE': 7
'KWRANGE': 8
'STRTYPE': 9
'ITYPE': 10
'FTYPE': 11
'BTYPE': 12
'CTEFLOAT': 13
'CTEINT': 14
'LTBOOL': 15
'OPREL': 16
'OPEQ': 17
'OPAT': 18
'OPMBR': 19
'OPAD': 20
'OPML': 21
'OPCONJ': 22
'OPDISJ': 23
'OPNEG': 24
'ST': 25
'CLN': 26
'PARST': 27
'SQBRST': 28
'CLBRST': 29
'PAREND': 30
'SQBREND': 31
'CLBREND': 32
'LTSTRING': 33
'KWRETURN': 34
'ID': 35