Autor: Mustafa Kuru
Mtrnr:0227482
Kennzahl:534
E-Mail:mustafakuru@msn.com

The Turing Programming Language

Inhaltsverzeichnis

1.Die Turing Programmiersprache
2.Historische Entwicklung von Turing
3.Überblick der Turing Programmiersprache
4.Grundlegende Symbole der Turing Sprache
5.Datentype und Erklärungen
6.Designziele der Turing Programmiersprache
7.Das Programm
8.Zusammenfassende Definition von Turing

1.Die Turing Programmiersprache

Turing ist entworfen, um eine universelle Programmiersprache zu sein . Turing ist eine Sprache die entwickelt worden ist um unabhängig des bestimmten Computers zu sein. Man sagt dass Turing ein verhältnismäßig einfaches Programm zu erlernen ist. Da es eine problemorientierte Sprache ist, wird es mit Problemen, numerischen Berechnungen wie Auftritt in den wissenschaftlichen Anwendungen einer Technik sowie mit der alphabetischen Manipulation der Informationen betroffen, die durch Geschäft und humanistische Anwendungen erfordert wird. Außerdem kann es für Programmiersystem-Software verwendet werden. Es ist eine verfahrensorientierte Sprache die bedeutet, dass sie ausdrücklichen Algorithmen entworfen war.

2.Historische Entwicklung der Turing Programmierung

Turing ist eine Programmiersprache wie Pascal, es hat sich von Richard C. Holt und James R. Cordy an der Universität von Toronto, Canada entwickelt, um über die Grundmodelle der Informatik zu unterrichten (1982).Turing ist ein Nachkommen von Euclid, mit den Eigenschaften der sauberen Syntax und der exakten Maschinen- Unabhängigkeitssemantik. Es wird hauptsächlich als eine unterrichtende Sprache „Orientiertes Turing und Turing Plus“ und als eine Programmierungsvariante verwendet.

3.Überblick der Turing Programmiersprache

Turing kann für ein Superpascal gehalten werden, dadurch dass es im wesentlichen alle Eigenschaften von Pascal enthält. Beim Hinzufügen der erforderlichen Eigenschaften wie dynamische Reihen, Module und Verändern der Länge der Zeichenketten.
Ein Turing Programm besteht aus Aussagen und Erklärungen und diese enthalten Ausdrücke. Aussagen inkludieren Anweisungen zu den Variablen, Schleifen bei „if“ und „case“ Aussagen. Erklärungen werden verwendet um Einzelteile wie Variablen als Subprogramms zu nennen. Ausdrücke in Turing sind ganz ähnlich wie die im Pascal mit den folgenden Bemerkungswerten.
Ausnahmen: Turing muss mehr Operatoren **(Exponentiation) und =(Implikation) haben.
Die Vorrangsrichtlinien erlauben bestimmte Ausdrücke die von Pascal verboten werden. z.B. in Turing kann man „i=1 oder J=1“schreiben…

4.Grundlegende Symbole der Turing Sprache

Jede mögliche Sprache besteht aus Wörtern und die Wörter sind gebildete Symbole, dass man Buchstaben benannt.
Die Buchstaben sind;
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
Kleinbuchstaben sind beim Turing äquivalent.
Die Stellen sind 0 1 2 3 4 5 6 7 8 9
Sonderzeichen: + - * / () = . " ` [ ] ; {} ! ~ |

4.1 Zahlen

In Turing werden Zahlen wie 2.3.512.809 und 46281 Ganzzahl Konstante genannt. Turing Compiler hat eine Strecke der Ganzzahlen einschließlich; von -2.147.483.647 bis 2.147.483.647.
Eine Ganzzahl-Konstante darf nicht ein Dezimalkomma haben.
Wenn Sie große Zahl haben, wie 635.642.000 die sie als $6.35642*10^{2*2*2}$ in Turing schreiben können.

4.2 Zeichenfolgen

Zeichenfolgen können aus Buchstaben bestehen. Die können Buchstaben, Stellen, Sonderzeichen und Ausdrückliche Konstanten sein. Z.B. „Bill Jones“„Technische Universität“ oder „x=“.
Wenn man die Resultate der Berechnungen eines Computers anzeigt, wünscht man die beschrifteten Resultate. Diese Zeichenketten, die in den Preisangabemarkierungen umgeben werden, werden ausdrückliche Zeichenkettekonstanten genannt.

4.3 Ausdrücke

Vor allem sind die Grundmodelle bei der Programmierung eines Ausdruckes alles 32,5,1e2, und 58,1e6 sind Ausdrücke. (32) und (61e2) sind auch Ausdrücke. Diese Ausdrücke werden arithmetische Ausdrücke genannt .
Für Vermehrungen benutzt man das Asterix (*) weil es kein Zeitzeichen gibt.

5.Datentype und Erklärungen

Integers, reelle - Zahlen, booleans, enumerated types, sets,
arrays, records.
Zeiger und Ansammlungen mit new und free Aussagen.
Anschlüsse (sichere Varianteaufzeichnungen) mit „tag„aussage.
Unterschiedliche Länge Zeichenketten.
Genannte Arten (wie im Pascal).
Operatoren: f, -, l , /, div, mod, ‘* (exponentiation), C, >, <=,
>=, =, not=, not, and, or, => (Implikation), in, not in.

5.1 Daten Erklärungen

var und const Erklärungen
bind Erklärung (Wiedereinbau von Pascal von with)

5.2 Input-Output

get,put Aussagen wird mit Formatierung und mit and-of-file entdeckt.
öffnen Sie und schließen Sie Verfahren, um Akten oder Befehl Zeilenposition namentlich zugänglich zu machen
open und close Verfahren wird verwendet, um Akten oder Befehl Zeilenposition namentlich zugänglich zu machen.

5.3 Steurung der Konstruktion

if Aussagen wird mit elsif und else Klauseln gesteuert.
case Aussagen mit anderer Klausel

loop und for Aussagen mit Ausgängen

5.4 Unterprogramme und Module

Prozedure mit return Aussagen
Funktion mit result Aussagen und mit Skalar oder nonscalar Resultat,
Dynamische Reihe und Zeichenkette Parameter.
Unterprogramme als Parameter.
Module mit Import und Export Tarife für Informationsverstecken.

5.5 Behauptungen

assert, pre, post, invariant (Korrektheitsanforderungen gebend).

5.6 Vorbestimmte Unterprogramme

Mathematische Funktionen: abs, max, min, sign, sqrt, sin, cos,
nrctan, sind, cosd, arctand, In, exp.
Transfer Funktionen Type: ceil, round, infreal, chr, ord, intstr,
strint, erealstr, frealstr, realstr, strreal.
Erzeugung der gelegentlichen Zahl: rand, randint, randomize,
randnext. randseed.

6.Designziele für Turing Programmiersprache

6.1 Allgemeinheit

Turing ist entworfen, um eine universelle Programmiersprache zu sein. Dieses Designziel bedeutet dass Turing die Sprache der Wahl für eine breite Kategorie Anwendungen möglicherweise sein sollte.

6.2 Mühelosigkeit des Lernens

Turing ist entworfen um Mühelos zu sein, zum erlernen. Dieses Designziel verlangt Einfachheit und Bequemlichkeit des Gebrauches. Es erfordert Spracheigenschaften die kurzen freien Erklärungen zu haben.

6.3 Zuverlässigkeit

Turing ist entworfen, um Zuverlässigkeit zu fördern. Dieses Designziel bedeutet, das Turing dem Programmierer in sich entwickelnder - Software helfen sollte, die zuverlässig seine spezifizierten Zwecke erreicht. Verständlichkeit erhöht Zuverlässigkeit

6.4 Mathematische Präzision
Turing ist entworfen, um eine mathematisch exakte (formale) Definition zu haben. Eine Antriebskraft hinter Design von Turing war die Richtlinie. Alle Resultate folgen von der Sprachenspezifikation.

6.5 Leistunfähigkeit
Turing ist für Leistungsfähigkeit bestimmt. Dieses Designziel bedeutet dass Turing Programme so klein und schnell wie möglich sein sollen. Wie in den Maschinennahen Programmiersprachen wie C.

6.6 Globale Ziele
Coherence: Die Sprache sollte in ein angenehmes Wesen zusammenpassen.
Enjoyment: Leute sollten mit der Sprache Spaß haben.
Acceplability: Leute sollten bereit sein die Sprache zu versuchen.

7.Das Programm
1.Programm
Das komplette Programm zum drucken „hallo“ ist

```
 put” Hello”
                                        %Eine Reihenfolge von Hello und von
                    Goodbye auf  unterschiedlichen Linien ausgeben, soll das
                    komplette Turing Programm
loop
   put “Hello”
   put “Goodbye”
 endloop
 %es ist, zum des Äquivalents im Pascal zu  betrachten ein  lehrreiches:
program test (output);
        begin
           while to do
                begin
                    writeln(“Hello”);
                  writeln(“Goodbye”)
                end
        end .
```

2.Programm

Ist hier ein komplettes Programm Turing, das nach dem zufall farbige Sterne in einem Nachthimmel zeichnet.

```
var x, y, clr : int                          % The location and color of  the star
   drawfillbox (0, 0, maxx, maxy, black)     % Make the window black
   drawline (0, 100, maxx, 100, white)       % Draw the horizon
   for i : 1 .. 40
      randint (x, 0, maxx)                    % Set the x position
      randint (y, 100, maxy)                  % Set the y position
      randint (clr, 0, maxcolor)              % Set the color
      drawfillstar (x, y, x + 20, y + 20, clr)   % Draw the star
   end for
```

8.Zusammenfassende Definition von Turing

Eins der Hauptdesignziele von Turing war in dieser Sprache, das formale Definition zugänglich sein sollte. Diese Spracheigenschaften wurden für die Überprüfung entworfen, ob vorhandene Formalisierung einer Sprache fähig ist, es zu ermöglichen dass die Produktion zur Formalen Definition angewendet wird, die im wesentlichen alle Aspekte der Sprache spezifiziert.

Literaturliste:

1. R.C. Holt
J.N.P. Hume
“Introduction to Computer Science using the Turing Programming Language”

2. Richard C. Holt
Philip A. Matthews
J. Alan Rosselet
James R. Cordy
“The Turing Programming Language Design and Definition”

3.
http://portal.acm.org/

4.
http://de.wikipedia.org/wiki/