TCVN TIÊU CHUẨN QUỐC GIA NATIONAL STANDARD

TCVN 7563-15 : 2009
ISO/IEC 2382-15 : 1995

Xuất bản lần 1
First Edition

CÔNG NGHỆ THÔNG TIN - TỪ VỰNG
PHẦN 15: NGÔN NGỮ LẬP TRÌNH

INFORMATION TECHNOLOGY – VOCABULARY
PART 15: PROGRAMMING LANGUAGES

HÀ NỘI - 2009

Mục lục **Trang**

Lời nói đầu 5

Mục 1: Khái quát 7

1.1 Phạm vi áp dụng 7

1.2 Tài liệu viện dẫn 7

1.3 Nguyên lý và quy tắc 7

Mục 2: Các thuật ngữ và các định nghĩa 10

15 Ngôn ngữ lập trình 10

15.01 Đơn vị từ vựng 10

15.02 Khai báo 13

15.03 Đối tượng dữ liệu 16

15.04 Kiểu dữ liệu 20

15.05 Câu lệnh và biểu thức 26

15.06 Phần chương trình 32

15.07 Tác vụ 38

15.08 Thi hành 39

15.09 Lập trình hướng đối tượng 40

15.10 Tính năng và đặc điểm 41

Content **Page**

Foreword 6

Section 1: General 7

1.1 Scope 7

1.2 Normative reference 7

1.3 Principles and rules followed 7

Section 2: Terms and definitions 10

15 Programming languages 10

15.01 Lexical tokens 10

15.02 Declarations 13

15.03 Data objects 16

15.04 Data types 20

15.05 Statements and expressions 26

15.06 Parts of programs 32

15.07 Tasks 38

15.08 Execution 39

15.09 Object-oriented programming 40

15.10 Features and characteristics 41

Lời nói đầu

TCVN 7563–15 : 2008 hoàn toàn tương đương với ISO/IEC 2382-15 : 1995.

TCVN 7563–15: 2008 do Ban Kỹ thuật Tiêu chuẩn quốc gia TCVN/TC 154 "*Quá trình, các yếu tố dữ liệu và tài liệu trong thương mại, công nghiệp và hành chính*" biên soạn, Tổng cục Tiêu chuẩn Đo lường Chất lượng đề nghị, Bộ Khoa học và Công nghệ công bố.

Foreword

National Standard **TCVN 7563-15 : 2009** is identical to International Standard ISO/IEC 2382-15 : 1995.

National Standard **TCVN 7563-15 : 2009** was prepared by National Technical Committee TCVN/JTC 1, *Information Technology*, approved by Directorate for Standards, Metrology and Quality, and published by Ministry of Science and Technology.

Công nghệ thông tin - Từ vựng
Phần 15: Ngôn ngữ lập trình
Information technology - Vocabulary
Part 15: Programming languages

Mục 1: Khái quát

1.1 Phạm vi áp dụng

Tiêu chuẩn này nhằm tạo thuận lợi cho việc truyền thông quốc tế trong công nghệ thông tin. Tiêu chuẩn trình bày bằng hai ngôn ngữ Việt-Anh các thuật ngữ và định nghĩa về những khái niệm được lựa chọn liên quan đến lĩnh vực công nghệ thông tin và xác định những quan hệ giữa các mục.

Để tạo thuận lợi cho việc dịch thuật sang các ngôn ngữ khác, các định nghĩa ở đây được biên soạn sao cho trong chừng mực có thể tránh khỏi mọi dị biệt của một ngôn ngữ.

Tiêu chuẩn này định nghĩa các khái niệm có liên quan đến đồ họa máy tính.

1.2 Tài liệu viện dẫn

ISO/IEC 2382-1:1993, Công nghệ thông tin - Từ vựng - Phần 1: Thuật ngữ cơ bản.

ISO 2382-2:1976, Xử lý dữ liệu - Từ vựng - Phần 02: Thao tác lôgic và số học.

ISO/IEC 2382-7:1991, Công nghệ thông tin - Từ vựng - Phần 7: Lập trình máy tính.

1.3 Nguyên lý và quy tắc

1.3.1 Định nghĩa một mục

Mục 2 gồm một số mục. Mỗi mục gồm có một tập hợp các phần tử cơ bản bao hàm một số hiệu chỉ mục, một thuật ngữ hoặc một vài thuật ngữ đồng nghĩa, và một mệnh đề định nghĩa một khái niệm. Thêm vào đó, một mục có thể bao hàm các ví dụ, chú thích hoặc minh họa nhằm tạo thuận lợi cho việc thông hiểu khái niệm.

Đôi khi, cùng một thuật ngữ có thể được định nghĩa trong các mục khác nhau, hoặc hai hoặc nhiều hơn

Section 1: General

1.1 Scope

This Standard is intended to facilitate international communication in information technology. It presents, in two languages, terms and definitions of selected concepts relevant to the field of information technology and identifies relationships among the entries.

In order to facilitate their translation into other languages, the definitions are drafted so as to avoid, as far as possible, any peculiarity attached to a language.

This Standard defines concepts related to reliability, maintainability, and availability.

1.2 Normative reference

ISO/IEC 2382-1:1993, Information technology - Vocabulary - Part 1: Fundamental terms.

ISO 2382-2:1976, Data processing - Vocabulary - Part 02: Arithmetic and logic operations.

ISO/IEC 2382-7:1991, Information technology - Vocabulary - Part 7: Computer programming.

1.3 Principles and rules followed

1.3.1 Definition of an entry

Section 2 comprises a number of entries. Each entry consists of a set of essential elements that includes an index number, one term or several synonymous terms, and a phrase defining one concept. In addition, an entry may include examples, notes or illustrations to facilitate understanding of the concept.

Occasionally, the same term may be defined in different entries, or two or more concepts may be

hai khái niệm có thể được định nghĩa bởi một mục, như đã mô tả tương ứng trong 1.3.5 và 1.3.8.

Các thuật ngữ khác như từ vựng, khái niệm, thuật ngữ, và định nghĩa được sử dụng trong tiêu chuẩn này đã được định nghĩa trong ISO 1087.

1.3.2 Tổ chức của một mục

Mỗi mục bao gồm các phần tử cơ bản được định nghĩa trong 1.3.1 và các phần tử được bổ sung nếu cần thiết. Mục đó có thể bao gồm các phần tử dưới đây theo thứ tự như sau:

a) Số hiệu chỉ mục (chung cho mọi ngôn ngữ sử dụng khi công bố phần này của tiêu chuẩn) ;

b) Thuật ngữ hoặc thuật ngữ được ưu tiên chung trong ngôn ngữ. Sự vắng mặt của một thuật ngữ được ưu tiên chung cho khái niệm đó trong ngôn ngữ sử dụng sẽ kí hiệu bởi 5 chấm (.....) ; một dòng các chấm có thể dùng để chỉ báo một từ cần chọn cho mỗi trường hợp cụ thể trong một thuật ngữ;

c) Thuật ngữ được ưu tiên trong một quốc gia cụ thể (được xác định theo các quy tắc của TCVN 7217) ;

d) Viết tắt của thuật ngữ ;

e) (Các) thuật ngữ đồng nghĩa được phép dùng ;

f) Văn bản của định nghĩa (xem 1.3.4) ;

g) Một hoặc một số ví dụ với tiêu đề "VÍ DỤ";

h) Một hoặc một số chú thích đặc tả các trường hợp riêng trong lĩnh vực ứng dụng các khái niệm với tiêu đề "CHÚ THÍCH" ;

i) Một hình ảnh, một biểu đồ, hoặc một bảng có thể dùng chung cho vài mục.

1.3.3 Phân loại mục

Một chuỗi số gồm hai chữ số được ấn định cho mỗi phần của bộ tiêu chuẩn này, bắt đầu là 01 cho "Các thuật ngữ căn bản".

Các mục được phân loại theo các nhóm, mỗi nhóm được ấn định một chuỗi số gồm 4 chữ số, trong đó hai chữ số đầu tiên dùng để chỉ phần của bộ tiêu chuẩn này.

Mỗi mục được ấn định một số chỉ mục gồm 6 chữ số, trong đó 4 chữ số đầu tiên dùng để chỉ phần của bộ tiêu chuẩn này và chỉ nhóm của mục. Những số trên được ấn định cho các hợp phần,

covered by one entry, as described in 1.3.5 and 1.3.8 respectively.

Other terms such as vocabulary, concept, term, and definition are used in This Standard with the meaning defined in ISO 1087.

1.3.2 Organization of an entry

Each entry contains the essential elements defined in 1.3.1 and, if necessary, additional elements. The entry may contain the following elements in the following order:

a) an index number (common for all languages in which This Standard is published) ;

b) the term or the generally preferred term in the language. The absence of a generally preferred term for the concept in the language is indicated by a symbol consisting of five dots (.....); a row of dots may be used to indicate, in a term, a word to be chosen in each particular case ;

c) the preferred term in a particular country (identified according to the rules of ISO 3166) ;

d) the abbreviation for the term ;

e) permitted synonymous term(s);

f) the text of the definition (see 1.3.4);

g) one or more examples with the heading "Example(s)";

h) one or more notes specifying particular cases in the field of application of the concepts with the heading "NOTE(S)";

i) a picture, a diagram, or a table which could be common to several entries.

1.3.3 Classification of entries

A two-digit serial number is assigned to each part of ISO/IEC 2382, beginning with 01 for "Fundamental terms".

The entries are classified in groups to each of which is assigned a four-digit serial number; the first two digits being those of the part of ISO/IEC 2382.

Each entry is assigned a six-digit index number; the first four digits being those of the part of ISO/IEC 2382 and the group. To show the relationship between versions of ISO/IEC 2382 in

các nhóm và các mục một cách giống nhau để các phiên bản của tiêu chuẩn này được nhất quán trong mọi ngôn ngữ sử dụng.

1.3.4 Lựa chọn các thuật ngữ và cách diễn đạt các định nghĩa

Việc lựa chọn các thuật ngữ và cách diễn đạt các định nghĩa, trong mức độ có thể, đã tuân theo cách sử dụng được thiết lập. Những nơi có mâu thuẫn đã được giải quyết thỏa thuận theo đa số phiếu bầu.

1.3.5 Đa nghĩa

Khi một thuật ngữ cho trước có nhiều nghĩa trong một ngôn ngữ làm việc, thì mỗi nghĩa được đưa vào một mục riêng nhằm tạo thuận lợi cho việc dịch thuật sang các ngôn ngữ khác.

1.3.6 Các viết tắt

Như đã nêu trong 1.3.2, các viết tắt hiện sử dụng chỉ được đặt ra cho một số thuật ngữ. Các viết tắt như vậy không được sử dụng trong văn bản của các định nghĩa, ví dụ hoặc chú thích.

1.3.7 Sử dụng dấu ngoặc đơn

Trong một số thuật ngữ, một hoặc nhiều từ in kiểu chữ đậm được đặt giữa các dấu ngoặc đơn. Những từ này là bộ phận của một thuật ngữ đầy đủ, nhưng có thể lược bỏ chúng khi sử dụng thuật ngữ rút gọn trong một ngữ cảnh kĩ thuật rõ ràng. Trong văn bản của một định nghĩa, ví dụ hoặc chú thích khác của tiêu chuẩn này, một thuật ngữ như vậy chỉ được sử dụng dưới dạng đầy đủ của nó.

Trong một số mục, các thuật ngữ được theo sau bởi các từ trong ngoặc đơn in với kiểu chữ thường. Những từ này không phải là bộ phận của một thuật ngữ nhưng nêu ra các hướng dẫn để sử dụng thuật ngữ đó, lĩnh vực áp dụng cụ thể hoặc dạng ngữ pháp của thuật ngữ đó.

1.3.8 Sử dụng dấu ngoặc vuông

Khi nhiều thuật ngữ có quan hệ mật thiết có thể được xác định bởi các văn bản chỉ khác nhau một vài từ, những thuật ngữ này và các định nghĩa của chúng sẽ được nhóm thành một mục đơn. Những từ cần thay thế để có các ý nghĩa khác nhau sẽ được đặt trong dấu ngoặc vuông, tức [ ], trong cùng thứ tự như trong thuật ngữ và trong định nghĩa đó. Để xác định rõ ràng các từ cần

various languages, the numbers assigned to parts, groups, and entries are the same for all languages.

1.3.4 Selection of terms and wording of definitions

The selection of terms and the wording of definitions have, as far as possible, followed established usage. Where there were contradictions, solutions agreeable to the majority have been sought.

1.3.5 Multiple meanings

When, in one of the working languages, a given term has several meanings, each meaning is given a separate entry to facilitate translation into other languages.

1.3.6 Abbreviations

As indicated in 1.3.2, abbreviations in current use are given for some terms. Such abbreviations are not used in the texts of the definitions, examples or notes.

1.3.7 Use of parentheses

In some terms, one or more words printed in bold typeface are placed between parentheses. These words are part of the complete term, but they may be omitted when use of the abridged term in a technical context does not introduce ambiguity. In the text of another definition, example, or note of ISO/IEC 2382, such a term is used only in its complete form.

In some entries, the terms are followed by words in parentheses in normal typeface. These words are not a part of the term but indicate directives for the use of the term, its particular field of application, or its grammatical form.

1.3.8 Use of brackets

When several closely related terms can be defined by texts that differ only in a few words, the terms and their definitions are grouped in a single entry. The words to be substituted in order to obtain the different meanings are placed in brackets, i.e. [ ], in the same order in the term and in the definition. To clearly identify the words to be substituted, the last word that according to the

thay thế, từ cuối cùng mà theo quy tắc nói trên có thể đặt trước dấu ngoặc vuông mở, sẽ được đặt trong dấu ngoặc này ở chỗ bất kỳ có thể, và lặp lại đối với mỗi từ khác.

1.3.9 Sử dụng các thuật ngữ được in theo kiểu chữ nghiêng trong các định nghĩa và việc sử dụng dấu hoa thị

Một thuật ngữ in kiểu chữ nghiêng trong một định nghĩa, ví dụ, hoặc chú thích, sẽ được định nghĩa trong một mục khác thuộc tiêu chuẩn này, mà có thể trong một hợp phần khác. Tuy nhiên, thuật ngữ đó chỉ in kiểu chữ nghiêng khi xuất hiện lần đầu trong mỗi mục.

Kiểu chữ nghiêng cũng được sử dụng cho các dạng ngữ pháp khác của một thuật ngữ, ví dụ danh từ số nhiều và động tính từ.

Các dạng cơ sở của tất cả các thuật ngữ in kiểu chữ nghiêng tại tiêu chuẩn này được liệt kê trong bảng chỉ mục ở cuối tiêu chuẩn (xem 1.3.11).

Dấu hoa thị dùng để tách các thuật ngữ in kiểu chữ nghiêng khi có hai thuật ngữ như thế được tham chiếu trong các mục riêng và đi theo sát nhau (hoặc chỉ được tách bởi dấu ngữ pháp).

Các từ hoặc thuật ngữ in kiểu chữ thường sẽ được hiểu như đã xác định trong các từ điển hiện hành hoặc các bộ từ vựng kĩ thuật chính thức.

1.3.10 Chính tả

Trong phiên bản tiếng Anh của tiêu chuẩn này, các thuật ngữ, định nghĩa, ví dụ và chú thích đều đánh vần theo kiểu chính tả được ưu tiên ở Mỹ. Các kiểu chính tả khác cũng có thể được sử dụng mà không trái với tiêu chuẩn này.

1.3.11 Tổ chức chỉ mục theo thứ tự ABC

Trong mỗi ngôn ngữ sử dụng sẽ có một chỉ mục xếp theo thứ tự ABC ở cuối hợp phần. Chỉ mục này gồm mọi thuật ngữ được định nghĩa trong hợp phần. Những thuật ngữ đa từ sẽ xuất hiện theo thứ tự ABC dưới mỗi từ khóa của chúng.

Mục 2: Các thuật ngữ và các định nghĩa

15 Ngôn ngữ lập trình

15.01 Đơn vị từ vựng

15.01.01

above rule could be placed in front of the opening bracket is, wherever possible, placed inside the bracket and repeated for each alternative.

1.3.9 Use of terms printed in italic typeface in definitions and the use of an asterisk

A term printed in italic typeface in a definition, an example, or a note is defined in another entry in ISO/IEC 2382, which may be in another part. However, the term is printed in italic typeface only the first time it occurs in each entry.

Italic typeface is also used for other grammatical forms of a term, for example, plurals of nouns and participles of verbs.

The basic forms of all terms printed in italic typeface which are defined in this part of ISO/IEC 2382 are listed in the index at the end of the part (see 1.3.11).

An asterisk is used to separate terms printed in italic typeface when two such terms are referred to in separate entries and directly follow each other (or are separated only by a punctuation mark).

Words or terms that are printed in normal typeface are to be understood as defined in current dictionaries or authoritative technical vocabularies.

1.3.10 Spelling

In the English language version of This Standard, terms, definitions, examples, and notes are given in the spelling preferred in the USA. Other correct spellings may be used without violating This Standard.

1.3.11 Organization of the alphabetical index

For each language used, an alphabetical index is provided at the end of each part. The index includes all terms defined in the part. Multiple-word terms appear in alphabetical order under each of their key words.

Section 2: Terms and definitions

15 Programming languages

15.01 Lexical tokens

15.01.01

phần tử từ vựng

đơn vị từ vựng

một chuỗi gồm một hoặc nhiều ký tự trong bảng chữ cái của ngôn ngữ lập trình, thường thể hiện đơn vị ý nghĩa căn bản.

VÍ DỤ: Chữ 2G5 hoặc một định danh last_name (tên cuối cùng) trong Pascal.

15.01.02

cấu trúc ngôn ngữ

Phần chương trình được phép về cú pháp có thể hình thành từ một hoặc nhiều đơn vị từ vựng phù hợp với các quy tắc ngôn ngữ lập trình.

15.01.03

định danh (trong ngôn ngữ lập trình)

Đơn vị từ vựng để đặt tên cấu trúc ngôn ngữ.

VÍ DỤ: Tên các biến, mảng, bản ghi, nhãn, thủ tục v.v....

CHÚ THÍCH - Định danh thường bao gồm một chữ cái và các chữ cái, chữ số hoặc ký tự khác theo sau một cách tùy chọn.

15.01.04

định danh định trước

Định danh được định nghĩa như một phần của ngôn ngữ lập trình.

VÍ DỤ: Từ dành riêng.

CHÚ THÍCH - Nếu định danh định trước không dành riêng thì khai báo có sử dụng định danh phải định nghĩa lại ý nghĩa của định danh đó trong phạm vi của khai báo.

15.01.05

từ dành riêng

Định danh định trước không thể định nghĩa lại bởi lập trình viên.

CHÚ THÍCH - Không phải tất cả ngôn ngữ lập trình đều có từ dành riêng.

lexical token

lexical element

lexical unit

A string of one or more characters of the alphabet of a programming language that, by convention, represents an elemental unit of meaning.

Examples: A literal such as 2G5 or an identifier such as last_name in Pascal.

15.01.02

language construct

A syntactically allowable part of a program that may be formed from one or more lexical tokens in accordance with the rules of a programming language.

15.01.03

identifier (in programming languages)

A lexical token that names a language construct.

Examples: The names of variables, arrays, records, labels, procedures, etc.

NOTE - An identifier usually consists of a letter optionally followed by letters, digits, or other characters.

15.01.04

predefined identifier

An identifier that is defined as part of a programming language.

Example: A reserved word.

NOTE - If a predefined identifier is not reserved, then a declaration using that identifier redefines its meaning for the scope of the declaration.

15.01.05

reserved word

A predefined identifier that cannot be redefined by a programmer.

NOTE - Not all programming languages have reserved words.

15.01.06

dấu phân cách (trong ngôn ngữ lập trình)

dấu phân tách (không sử dụng theo nghĩa này)

Đơn vị từ vựng biểu thị sự bắt đầu hoặc kết thúc của một đơn vị từ vựng khác hoặc của một chuỗi ký tự được xem như đơn vị cú pháp.

CHÚ THÍCH

1. Các ký tự cụ thể hoặc các từ dành riêng có thể dùng như dấu phân cách.

2. Ngược lại với dấu phân tách.

15.01.07

dấu phân tách

Dấu phân cách ngăn các đơn vị từ vựng liền kề hoặc các đơn vị cú pháp để tránh hiểu chúng như một mục đơn.

VÍ DỤ: Ký tự dấu cách hoặc bộ tạo định dạng.

CHÚ THÍCH - Ngược lại với dấu phân cách.

15.01.08

làm quá tải

Gán nhiều nghĩa cho một đơn vị từ vựng.

VÍ DỤ: Đơn vị từ vựng "+" có thể có nghĩa là phép cộng số nguyên, phép cộng số thực, phép hội, ghép nối v.v...

15.01.09

làm thành nghĩa

Hành động định rõ cấu trúc ngôn ngữ của vài ngôn ngữ với cùng trình tự các đơn vị từ vựng được đề cập bởi lần xuất hiện nào đó trong chương trình.

15.01.10

nhãn (trong ngôn ngữ lập trình)

Định danh vị trí trong chương trình.

CHÚ THÍCH

1. Nhãn thường dùng để tham chiếu câu lệnh.

2. Trong BASIC, số hiệu dòng có thể dùng như nhãn, nhưng không phải là đích khi truyền.

15.01.06

delimiter (in programming languages)

separator (deprecated in this sense)

A lexical token that indicates the beginning or the end of another lexical token or of a character string considered as a syntactic unit.

NOTES

1. Special characters or reserved words may serve as delimiters.

2. Contrast with separator.

15.01.07

separator

A delimiter that prevents adjacent lexical tokens or syntactic units from being interpreted as a single item.

Examples: The space character or a format effector.

NOTE - Contrast with delimiter.

15.01.08

to overload

To assign more than one meaning to a lexical token.

Example: The lexical token "+" can mean integer addition, real addition, set union, concatenation, etc.

15.01.09

disambiguation

The action of determining which language construct, of several with the same sequence of lexical tokens, is referred to by a particular occurrence within a program.

15.01.10

label (in programming languages)

An identifier for a location in a program.

NOTES

1. A label is frequently used to refer to a statement.

2. In BASIC, a line number can serve as a label, but is not always the target of a transfer.

3. Trong Fortran, một nhãn có 5 chữ số đứng trước một câu lệnh có thể dùng để tham chiếu cho câu lệnh đó.

15.01.11

bình luận

chú giải

Cấu trúc ngôn ngữ được sử dụng riêng bao gồm văn bản không nhằm tác động đến việc thi hành chương trình.

VÍ DỤ: Việc giải thích cho người đọc; dữ liệu cho hệ thống tài liệu tự động.

15.02 Khai báo

15.02.01

khai báo

Cấu trúc ngôn ngữ rõ ràng đưa ra một hoặc nhiều định danh vào một chương trình và quy định cách thông dịch các định danh này.

VÍ DỤ: Khai báo các kiểu dữ liệu, tổ chức lưu trữ, các gói hoặc các tác vụ.

CHÚ THÍCH - Trong một vài ngôn ngữ lập trình, khai báo được coi là câu lệnh.

15.02.02

phần khai báo

đoạn dữ liệu

Phần chương trình bao gồm một hoặc nhiều khai báo.

CHÚ THÍCH - Trong COBOL, phần khai báo được gọi là "đoạn dữ liệu".

15.02.03

mặc định (tính từ)

Gắn liền với một thuộc tính, giá trị dữ liệu hoặc sự tùy chọn được thừa nhận khi không có quy định rõ ràng.

VÍ DỤ: Trong Fortran, quy ước đặt tên mặc định quy định rằng các tên bắt đầu với một trong các chữ cái từ I đến N biểu thị các biến kiểu số nguyên.

3. In Fortran, a label, consisting of up to five digits, that precedes a statement, may be used to refer to the statement.

15.01.11

comment

remark

A language construct exclusively used to include text that has no intended effect on the execution of the program.

Examples: An explanation to a human reader; data for an automatic documentation system.

15.02 Declarations

15.02.01

declaration

An explicit language construct that introduces one or more identifiers into a program and specifies how these identifiers are to be interpreted.

Examples: Declarations of data types, storage organization, packages, or tasks.

NOTE - In some programming languages, declarations are considered to be statements.

15.02.02

declarative part

data division

A portion of a program that consists of one or more declarations.

NOTE - In COBOL, a declarative part is called "data division".

15.02.03

default (adj.)

Pertaining to an attribute, data value, or option that is assumed when none is explicitly specified.

Example: In Fortran, the default naming convention specifies that names beginning with one of the letters I through N denote variables of integer type.

15.02.04

khai báo hàm ẩn

Khai báo do sự xuất hiện của định danh để chỉ rõ một đối tượng mà đặc điểm của đối tượng đó được xác định mặc định.

VÍ DỤ: Trong Pascal "đầu ra = văn bản".

15.02.05

định trước

cài sẵn

bản chất

Gắn liền với một cấu trúc ngôn ngữ được khai báo bởi việc định nghĩa ngôn ngữ lập trình đó.

VÍ DỤ: Chức năng định trước SIN trong PL/I, kiểu dữ liệu định trước INTEGER (số nguyên) trong Fortran.

15.02.06

phạm vi

phạm vi khai báo

Một phần chương trình mà trong đó khai báo là hợp lệ.

15.02.07

dữ liệu chia xẻ

Dữ liệu có thể được truy cập bởi hai hoặc nhiều môđun được thi hành một cách đồng thời hoặc không đồng thời.

VÍ DỤ: COMMON trong Fortran; "compool" trong một vài ngôn ngữ lập trình; EXTERNAL được gắn thẻ các biến đơn trong PL/I; dạng gói trong Ada.

15.02.08

phạm vi động

Phạm vi được tạo bởi việc kích hoạt các phần hoặc toàn bộ các môđun chứa khai báo được sử dụng bởi một môđun khác mà không có các khai báo này trong khoảng thời gian thực thi môđun sau đó.

15.02.04

implicit declaration

A declaration caused by the occurrence of an identifier that designates an object, whose characteristics are determined by default.

Example: In Pascal "output = text".

15.02.05

predefined

built-in

intrinsic

Pertaining to a language construct that is declared by the definition of the programming language.

Examples: The predefined function SIN in PL/I, the predefined data type INTEGER in Fortran.

15.02.06

scope

scope of a declaration

That portion of a program within which a declaration is valid.

15.02.07

shared data

Data that can be accessed by two or more modules that may be executed asynchronously or concurrently.

Examples: COMMON in Fortran; "compool" in some programming languages; PL/I single variables tagged EXTERNAL; a form of a package in Ada.

15.02.08

dynamic scope

The scope created by the activation of portions or all of the modules that contain declarations used by another module that lacks these declarations during the execution of the latter module.

15.02.09

phạm vi tĩnh

Phạm vi được định rõ bởi việc tìm kiếm mô đun phụ cận bên trong cùng tạo ra khai báo đó.

CHÚ THÍCH - Việc kiểm tra bàn làm việc của chương trình có khả năng tìm kiếm phạm vi tĩnh.

15.02.10

vùng khai báo

Phần chương trình bao gồm các khai báo.

15.02.11

cục bộ (tính từ)

Gắn liền với một cấu trúc ngôn ngữ có phạm vi chỉ nằm trong vùng khai báo mà nó được khai báo

15.02.12

toàn cục

Gắn liền với một cấu trúc ngôn ngữ có phạm vi nằm trong tất cả các mô đun của chương trình đó.

15.02.13

bên ngoài

Gắn liền với một cấu trúc ngôn ngữ được định nghĩa bên ngoài môđun mà nó được tham chiếu.

CHÚ THÍCH - Có thể yêu cầu một khai báo trong môđun để cung cấp tên và để chỉ rõ định nghĩa đầy đủ bên ngoài.

15.02.14

tĩnh (tính từ)

Gắn liền với các đối tượng tồn tại và giữ nguyên các giá trị của chúng thông qua việc thi hành toàn bộ chương trình.

VÍ DỤ: Biến của chương trình con được khai báo tĩnh vẫn giữ nguyên các giá trị của nó từ một câu lệnh thi hành đến câu lệnh thi hành tiếp theo.

15.02.15

động

Gắn liền với thuộc tính dữ liệu mà các giá trị của nó chỉ được thiết lập trong khoảng thời gian thi

15.02.09

static scope

The scope as determined by finding the innermost surrounding module in which the declaration is made.

NOTE - Desk checking of a program is sufficient for finding a static scope.

15.02.10

declarative region

A portion of a program consisting of declarations.

15.02.11

local (adj.)

Pertaining to a language construct that has a scope only within the declarative region in which it is declared.

15.02.12

global

Pertaining to a language construct that is within the scope of all modules of the program.

15.02.13

external

Pertaining to a language construct that is defined outside the module in which it is referenced.

NOTE - A declaration may be required within the module to provide a name and to indicate that the complete definition is external.

15.02.14

static (adj.)

Pertaining to objects that exist and retain their values throughout the execution of the entire program.

Example: A subprogram variable that has been declared static to retain its values from one execution to the next.

15.02.15

dynamic

Pertaining to a data attribute, whose values can only be established during the execution of all or

hành một phần hoặc toàn bộ chương trình.

VÍ DỤ: Chiều dài của một đối tượng dữ liệu có chiều dài có thể thay đổi là động.

15.02.16

vòng đời

Một phần khoảng thời gian thi hành mà trong thời gian đó một cấu trúc ngôn ngữ tồn tại.

15.02.17

tính hiện hữu (1)

Khả năng tạo một tham chiếu tới một cấu trúc ngôn ngữ nào đó tại vị trí cụ thể trong một môđun.

15.02.18

tính hiện hữu (2)

Phần chương trình mà trong đó có thể tạo ra một tham chiếu tới một cấu trúc ngôn ngữ cụ thể.

15.03 Đối tượng dữ liệu

15.03.01

cấu trúc dữ liệu

Mối quan hệ vật lý hoặc logic giữa các đơn vị dữ liệu và bản thân dữ liệu đó.

15.03.02

đối tượng dữ liệu (trong ngôn ngữ lập trình)

Phần tử cấu trúc dữ liệu như tệp, mảng hoặc toán hạng cần thiết để thi hành chương trình.

CHÚ THÍCH - Đối tượng dữ liệu có thể là hằng số hoặc biến số.

15.03.03

biến số

Được thiết lập bởi một khai báo hoặc khai báo hàm ẩn, gồm bốn phần: một định danh, một tập các thuộc tính dữ liệu, một hoặc nhiều địa chỉ và các giá trị dữ liệu, ở đây, mối quan hệ giữa các địa chỉ và giá trị dữ liệu có thể thay đổi.

CHÚ THÍCH - Trong một số ngôn ngữ lập trình, các địa chỉ có thể thay đổi, do đó, các giá trị dữ liệu tương ứng có thể thay đổi. Trong các ngôn ngữ lập trình khác, các địa chỉ là cố định, nhưng các giá trị dữ liệu tương ứng

part of a program.

Example: The length of a variable-length data object is dynamic.

15.02.16

lifetime

The portion of the execution duration during which a language construct exists.

15.02.17

visibility (1)

The ability to make a reference to a particular language construct at a specific place in a module.

15.02.18

visibility (2)

The portion of a program within which a reference can be made to a specific language construct.

15.03 Data objects

15.03.01

data structure

A physical or logical relationship among units of data and the data themselves.

15.03.02

data object (in programming languages)

An element of a data structure such as a file, an array, or an operand, that is needed for the execution of programs.

NOTE - A data object may be a constant or a variable.

15.03.03

variable

A quadruple, established by a declaration or an implicit declaration, that consists of an identifier, a set of data attributes, one or more addresses, and data values, where the relationship between the addresses and the data values may vary.

NOTE - In some programming languages, the addresses may vary, hence the associated data values may vary. In other programming languages, the addresses remain fixed, but the associated data values

có thể thay đổi trong khoảng thời gian thi hành.

15.03.04

giá trị dữ liệu

Phần tử của tập các đối tượng dữ liệu đã khai báo trong ngữ cảnh cụ thể được kết hợp với một cấu trúc ngôn ngữ như một biến số hoặc kiểu dữ liệu.

CHÚ THÍCH - Về nguyên tắc, trong toán học, giá trị dữ liệu phải phân biệt với "giá trị hàm", trong phép biểu diễn số, giá trị dữ liệu phải phân biệt với "giá trị vị trí".

15.03.05

hằng số

Được thiết lập bởi một khai báo hoặc một khai báo hàm ẩn, gồm bốn phần: một định danh, một tập các thuộc tính dữ liệu, một hoặc nhiều địa chỉ và một giá trị dữ liệu.

15.03.06

tập kết hợp

Tập hợp có cấu trúc của các thành phần, các phần tử ở đây có cùng cấu trúc dữ liệu hoặc có cấu trúc dữ liệu khác nhau. Các cấu trúc dữ liệu của tập hợp này cũng có thể là một phần cấu thành của kiểu phức hợp tương ứng.

15.03.07

giá trị tập kết hợp

Giá trị dữ liệu tương ứng với một tập kết hợp

15.03.08

mảng

Tập kết hợp là một trường hợp cụ thể của kiểu mảng và mỗi phần tử hoặc nhóm các phần tử thích hợp có thể được tham chiếu ngẫu nhiên và độc lập với phần tử hoặc nhóm các phần tử khác.

15.03.09

phần mảng

Một phần của mảng bao gồm các ô liên tục dọc theo mọi chiều.

may change during execution.

15.03.04

data value

An element of a declared set of data objects that, in a specific context, is associated with a language construct such as a variable or a data type.

NOTE - In principle, the data value should be distinguished from "function value" in mathematics, from "value of a number" and "position value" in numeric representation.

15.03.05

constant

A quadruple, established by a declaration or an implicit declaration, that consists of an identifier, a set of data attributes, one or more addresses, and only one data value.

15.03.06

aggregate

A structured collection of components, where the components may have the same or different data structure, and where the data structure of the collection itself may also be a constituent part of a corresponding composite type.

15.03.07

aggregate value

The data value associated with an aggregate.

15.03.08

array

An aggregate that is an instance of an array type and each element or appropriate group of elements in which may be referenced randomly and independently of the others.

15.03.09

array slice

slice

A portion of an array that consists of contiguous cells along any dimension.

CHÚ THÍCH - Trong Ada, một lát mảng cũng là một thao tác cơ bản.

15.03.10

phần biến thể

Một phần của bản ghi, gồm các đối tượng dữ liệu mà cấu trúc dữ liệu tương ứng của nó hoặc các kiểu dữ liệu đã khai báo có thể thay đổi.

CHÚ THÍCH - Có thể thay đổi cả số lượng các đối tượng dữ liệu và thành phần cấu tạo chúng.

15.03.11

bản ghi biến thể

Bản ghi gồm một phần biến thể.

CHÚ THÍCH - Bản ghi có thể gồm biệt thức để chỉ ra các kiểu dữ liệu trong phần biến thể.

15.03.12

biệt thức (danh từ)

Tham số - như cấu trúc ngôn ngữ chỉ ra cấu trúc dữ liệu được dùng trong một bản ghi biến thể cho trước.

15.03.13

tham số (trong ngôn ngữ lập trình)

Cấu trúc ngôn ngữ để truyền đối tượng dữ liệu hoặc giá trị dữ liệu giữa các mô đun.

15.03.14

tham số thực

đối số thực

Tham số như biểu thức, định danh hoặc cấu trúc ngôn ngữ khác, được sử dụng trong câu lệnh gọi hoặc thể hiện chung đối với việc kết hợp đối tượng dữ liệu với một khai báo tương ứng.

CHÚ THÍCH - Khai báo tương ứng được gọi là *tham số hình thức*.

15.03.15

tham số hình thức

đối số giả

Tham số, được xác định trong khai báo của mô đun nhất định, kết hợp với một *tham số thực* trong một câu lệnh gọi hoặc thể hiện chung.

NOTE - In Ada, an array slice is also a basic operation.

15.03.10

variant part

A part of a record, composed of data objects, whose corresponding data structures or declared data types may vary.

NOTE - Both the number of data objects and their composition may vary.

15.03.11

variant record

A record that contains a variant part.

NOTE - The record may contain discriminants to indicate the data types in the variant part.

15.03.12

discriminant (noun)

A parameter-like language construct that indicates the data structure to be used within a given variant record.

15.03.13

parameter (in programming languages)

A language construct for passing data objects or data values between modules.

15.03.14

actual parameter

actual argument

A parameter, such as an expression, identifier, or other language construct, used in a call or generic instantiation for association of a data object with a corresponding declaration.

NOTE - The corresponding declaration is called *formal parameter*.

15.03.15

formal parameter

dummy argument

A parameter, defined in the declaration of certain modules, that is associated with an *actual parameter* in a call or generic instantiation.

15.03.16

kết hợp tham số

Sự kết hợp giữa các *tham số hình thức* và *tham số thực* tương ứng với chúng trong câu lệnh gọi hoặc thể hiện chung.

15.03.17

thuộc tính dữ liệu

Đặc điểm định trước của kiểu dữ liệu, đối tượng dữ liệu, mô đun hoặc một vài cấu trúc ngôn ngữ khác.

VÍ DỤ: Kiểu số thực có thể có thuộc tính dữ liệu PRECISION với các giá trị dữ liệu SINGLE hoặc DOUBLE. Một tác vụ có thể có thuộc tính TERMINATED là TRUE nếu tác vụ kết thúc và các trường hợp khác là FALSE.

15.03.18

Hạn định tên

Hạn định

Phương thức tham chiếu các cấu trúc ngôn ngữ trong phạm vi một phần chương trình bởi tham chiếu tới phần chương trình đó và một định danh khai báo đối với cấu trúc ngôn ngữ trong phần đó.

VÍ DỤ: Sử dụng việc tham chiếu các thành phần bản ghi (B OF A trong COBOL), các thành viên của một thư viện, các cấu trúc ngôn ngữ trong một mô đun.

15.03.19

bí danh

Định danh thay thế cho một cấu trúc ngôn ngữ.

15.03.20

con trỏ (trong ngôn ngữ lập trình)

Đối tượng dữ liệu mà giá trị dữ liệu của nó là địa chỉ của một đối tượng dữ liệu khác.

CHÚ THÍCH - Xem Hình 1.

15.03.21

con trỏ rỗng

Con trỏ không chỉ rõ tới bất kỳ đối tượng dữ liệu nào.

CHÚ THÍCH - Phụ thuộc vào ngôn ngữ lập trình, con trỏ rỗng được thể hiện là "nil", "null", v.v...

15.03.16

parameter association

The association between *formal parameters* and their corresponding *actual parameters* in a call or generic instantiation.

15.03.17

data attribute

A predefined characteristic of a data type, data object, module, or some other language construct.

Examples: A real type may have the data attribute PRECISION with the data values SINGLE or DOUBLE. A task may have the data attribute TERMINATED that yields TRUE if the task is terminated, and FALSE otherwise.

15.03.18

name qualification

qualification

A means of referencing language constructs within the scope of a portion of a program by reference to that portion and an identifier declared for the language construct in that portion.

Example: Used for referencing record components (B OF A in COBOL), members of a library, language constructs in a module.

15.03.19

alias

An alternate identifier for a language construct.

15.03.20

pointer (in programming languages)

A data object whose data value is the address of another data object.

NOTE - See figure 1.

15.03.21

null pointer

A pointer that explicitly does not point to any data object.

NOTE - Depending on the programming language, the null pointer has a representation called "nil", "null", etc.

15.04 Kiểu dữ liệu

15.04.01 (17.05.08)

kiểu dữ liệu

Tập xác định các đối tượng dữ liệu của một cấu trúc dữ liệu quy định và tập các thao tác được phép, sao cho các đối tượng dữ liệu này hoạt động như các toán hạng trong thi hành thao tác.

VÍ DỤ: Kiểu số nguyên có cấu trúc đơn giản, mỗi lần xuất hiện thường là giá trị, là một cách biểu diễn của một thành phần trong dải quy định của toàn bộ các số của nó và các thao tác được phép bao gồm các thao tác số học thông thường trên số nguyên.

CHÚ THÍCH

1. Thuật ngữ "kiểu" có thể dùng thay cho "kiểu dữ liệu" khi đã rõ ràng.

2. Xem Hình 1.

15.04.02

kiểu dữ liệu tóm tắt

ADT (viết tắt)

Lớp các cấu trúc dữ liệu được mô tả bởi một danh sách các thao tác hoặc tính năng sẵn có trong các cấu trúc dữ liệu và các đặc tính hình thức của các thao tác này, với các giao diện phân tách với thực thi bên trong.

15.04.03

kiểu tóm lược

Kiểu dữ liệu của cấu trúc bên trong và các thao tác tương ứng có các giao diện được xác định công khai và phần mềm thực thi được xác định riêng.

15.04.04

kiểu vô hướng

kiểu đơn giản

Kiểu dữ liệu mà mỗi trường hợp cụ thể biểu thị vô hướng.

CHÚ THÍCH

1. Các *kiểu vô hướng* trong Pascal hoặc là kiểu thứ tự hoặc là

15.04 Data types

15.04.01 (17.05.08)

data type

datatype

A defined set of data objects of a specified data structure and a set of permissible operations, such that these data objects act as operands in the execution of any one of these operations.

Example: An integer type has a very simple structure, each occurrence of which, usually called value, is a representation of a member of a specified range of whole numbers and the permissible operations include the usual arithmetic operations on these integers.

NOTES

1. The term "type" may be used instead of "data type" when there is no ambiguity.

2. See figure 1.

15.04.02

abstract data type

ADT (abbreviation)

A class of data structures described by a list of operations or features available in the data structures and the formal properties of these operations, with the interfaces separated from the internal implementation.

15.04.03

encapsulated type

A data type with publicly defined interfaces and privately defined implementation of the internal structure and the associated operations.

15.04.04

scalar type

simple type

A data type, each instance of which represents a scalar.

NOTES

1. Pascal *scalar types* are either ordinal types or real

kiểu số thực. Các *kiểu vô hướng* trong Ada hoặc là kiểu rời rạc hoặc là kiểu số thực.

2. Xem Hình 1.

15.04.05

kiểu nguyên tử

Kiểu dữ liệu mà mỗi đối tượng dữ liệu gồm một giá trị dữ liệu không thể phân chia nhỏ.

15.04.06

kiểu logic

kiểu Boolean

Kiểu dữ liệu mà các đối tượng dữ liệu của nó chỉ thừa nhận các giá trị logic (thường là ĐÚNG hoặc SAI) và chỉ được tác dụng bởi các thao tác Boolean.

CHÚ THÍCH - Xem kiểu ký tự, *kiểu liệt kê*, *kiểu số nguyên*, *kiểu số thực*.

15.04.07

dải (trong ngôn ngữ lập trình)

khoảng độ (không sử dụng trong trường hợp này)

Tập các giá trị dữ liệu liền kề của *kiểu vô hướng*.

15.04.08

kiểu số thực

Kiểu dữ liệu mà mỗi đối tượng dữ liệu của nó biểu thị một số thực, có thể là số gần đúng.

VÍ DỤ: Số thập phân 0.1 trong hệ nhị phân có một số vô hạn các chữ số.

CHÚ THÍCH

1. Các *kiểu số thực* là các kiểu dấu phẩy cố định hoặc các kiểu dấu phẩy động.

2. Xem Hình 1.

15.04.09

kiểu dấu phẩy cố định

kiểu thập phân mặc nhiên

Kiểu số thực mà mỗi đối tượng dữ liệu của nó biểu thị một hệ biểu diễn dấu phẩy cố định.

CHÚ THÍCH - Xem Hình 1.

types. Ada *scalar types* are either discrete types or real types.

2. See figure 1.

15.04.05

atomic type

A data type, each data object of which consists of a single non-decomposable data value.

15.04.06

logical type

Boolean type

A data type whose data objects can assume only logical values (usually TRUE or FALSE) and can be operated on only by Boolean operators.

NOTE - See also *character type*, *enumeration type*, *integer type*, *real type*.

15.04.07

range (in programming languages)

span (không sử dụng trong trường hợp này)

A contiguous set of data values of a *scalar type*.

15.04.08

real type

A data type, each data object of which represents a real number, possibly by approximation.

Example: The decimal number 0.1 in the binary system has an infinite number of digits.

NOTES

1. Real types are either fixed-point types or floating-point types.

2. See figure 1.

15.04.09

fixed-point type

implied decimal type

A *real type*, each data object of which is expressed in a fixed-point representation system.

NOTE - See figure 1.

15.04.10

kiểu dấu phẩy động

Kiểu số thực mà mỗi đối tượng dữ liệu của nó biểu thị một hệ biểu diễn dấu phẩy động.

CHÚ THÍCH - Xem Hình 1.

15.04.11

kiểu thứ tự

kiểu rời rạc

Kiểu dữ liệu mà mỗi đối tượng dữ liệu của nó thể hiện một thành phần của một tập thứ tự có thể đếm.

CHÚ THÍCH

1. Các kiểu thứ tự trong Pascal là "enumerated", "char", " integer" và "Boolean". Các kiểu thứ tự trong Ada là các *kiểu số nguyên* hoặc *kiểu liệt kê*.

2. Xem Hình 1.

15.04.12

kiểu chỉ mục

Kiểu thứ tự mà mỗi đối tượng dữ liệu của nó thể hiện một chỉ số dưới của mảng.

15.04.13

kiểu số nguyên

Kiểu thứ tự mà mỗi đối tượng dữ liệu của nó mô tả một số nguyên trong một dãy quy định.

CHÚ THÍCH - Xem Hình 1.

15.04.14

kiểu liệt kê

kiểu được liệt kê

Kiểu thứ tự mà đối tượng dữ liệu của nó được liệt kê rõ ràng trong việc khai báo như một kiểu dữ liệu.

CHÚ THÍCH - Xem Hình 1.

15.04.15

kiểu số

Kiểu vô hướng mà mỗi đối tượng dữ liệu của nó mô tả một số nguyên hoặc khoảng xấp xỉ của số thực.

CHÚ THÍCH - Xem Hình 1.

15.04.10

floating-point type

A *real type*, each data object of which is expressed in a floating-point representation system.

NOTE - See figure 1.

15.04.11

ordinal type

discrete type

A data type, each data object of which represents a member of an ordered countable set.

NOTES

1. Pascal ordinal types are "enumerated", "char", "integer", and "Boolean". Ada ordinal types are either *integer types* or *enumeration types*.

2. See figure 1.

15.04.12

index type

An ordinal type, each data object of which represents a subscript of an array.

15.04.13

integer type

An ordinal type, each data object of which represents a whole number within a specified range.

NOTE - See figure 1.

15.04.14

enumeration type

enumerated type

An ordinal type whose data objects are explicitly enumerated in the declaration of such a data type.

NOTE - See figure 1.

15.04.15

numeric type

A *scalar type*, each data object of which represents an integer or an approximation of a real number.

NOTE - See figure 1.

15.04.16

kiểu ký tự

Kiểu thứ tự mà mỗi đối tượng dữ liệu của nó mô tả một ký tự.

CHÚ THÍCH - Xem Hình 1.

15.04.17

kiểu chuỗi

Kiểu dữ liệu mà mỗi đối tượng dữ liệu của nó là một chuỗi

CHÚ THÍCH - Xem Hình 1.

15.04.18

kiểu con trỏ

kiểu truy cập

Kiểu dữ liệu mà mỗi đối tượng dữ liệu của nó là một con trỏ.

CHÚ THÍCH - Xem Hình 1.

15.04.19

kiểu mảng

Kiểu phức hợp mà các thành phần của nó giống nhau.

CHÚ THÍCH

1. Các kiểu mảng có thể tổ chức và tham chiếu như là các thành phần được sắp xếp theo cột, dòng, v.v...

2. Xem Hình 1.

15.04.20

kiểu bản ghi

Kiểu phức hợp mà các thành phần của nó là kiểu trường hoặc là các kiểu bản ghi khác.

VÍ DỤ: Bản ghi nhân sự có thể bao gồm dữ liệu nhân sự sắp xếp như các trường hoặc các bản ghi phụ trong bản ghi nhân sự này.

CHÚ THÍCH

1. Kiểu bản ghi định nghĩa một tập các giá trị và các thao tác. Một ví dụ như kiểu bản ghi có thể gồm các giá trị bản thân chúng là các bản ghi.

2. Xem Hình 1.

3. Định nghĩa này đồng nhất với định nghĩa trong mục

15.04.16

character type

An ordinal type, each data object of which represents a character.

NOTE - See figure 1.

15.04.17

string type

A data type, each data object of which is a string.

NOTE - See figure 1.

15.04.18

pointer type

access type

A data type, each data object of which is a pointer.

NOTE - See figure 1.

15.04.19

array type

A composite type whose components are the same data type.

NOTES

1. Array types may be organized and referenced as if the components were arranged in columns, rows, etc.

2. See figure 1.

15.04.20

record type

A composite type whose components are field types or other record types.

Example: A personnel record may consist of personal data arranged as fields or sub-records within this personnel record.

NOTES

1. A record type defines a set of values and operations. An instance of such a record type may contain values which themselves are records.

2. See figure 1.

3. This definition is identical to the definition in entry

17.05.13 của ISO/IEC 2382-17. Ví dụ và chú ý đã được thêm vào.

15.04.21

kiểu bản ghi biến thể

Kiểu bản ghi có một phần biến thể chỉ rõ danh sách các thành phần thay thế.

15.04.22

kiểu phụ

Kiểu dữ liệu bắt nguồn từ kiểu dữ liệu khác bởi một hoặc nhiều ràng buộc trên kiểu dữ liệu khác đó.

15.04.23

kiểu cơ sở

kiểu máy chủ

kiểu cơ bản

Kiểu dữ liệu tạo ra một kiểu phụ.

CHÚ THÍCH - Ngược lại với kiểu cha.

15.04.24

ràng buộc

Sự thích ứng của một kiểu dữ liệu để giới hạn các thao tác hoặc dải của nó.

15.04.25

kiểu riêng

Trong một chương trình, kiểu dữ liệu mà cấu trúc, tập giá trị và các thao tác của nó đã được định nghĩa nhưng tính sẵn có của nó thì bị giới hạn cho các phần được phân quyền của chương trình đó.

VÍ DỤ: Trong Ada, phép gán, đẳng thức và bất đẳng thức sẵn có cho người sử dụng, ngoại trừ đối với mọi thao tác hoàn toàn sẵn dùng.

15.04.26

kiểu hạn chế

Kiểu riêng mà chỉ các thao tác hoặc thuộc tính dữ liệu được khai báo rõ ràng là sẵn có bên ngoài phần chương trình chứa *kiểu riêng* đó.

15.04.27

kiểu cha

Kiểu dữ liệu dùng như khuôn mẫu để tạo các kiểu

17.05.13 of ISO/IEC 2382-17. The example and notes have been added.

15.04.21

variant record type

A record type that has a variant part specifying alternative lists of components.

15.04.22

subtype

A data type derived from another data type by one or more constraints on that other data type.

15.04.23

base type

host type

underlying type

A data type from which a subtype is descended.

NOTE - Contrast with parent type.

15.04.24

constraint

An adaptation of a data type that restricts its range or operations.

15.04.25

private type

Within a program, a data type whose structure, set of values, and operations are defined but whose availability is restricted to privileged parts of that program.

Example: In Ada, only assignment, equality, and inequality are available to users, except for any operations explicitly made accessible.

15.04.26

limited type

A private type for which only explicitly declared operations or data attributes are available outside a part of a program in which it is contained.

15.04.27

parent type

A data type that serves as the template for

dữ liệu mới.

CHÚ THÍCH - Ngược lại với kiểu cơ sở, kiểu dẫn xuất.

15.04.28

kiểu dẫn xuất

Kiểu dữ liệu có các thao tác và giá trị dữ liệu là bản sao giống hệt của một kiểu cha hiện có.

CHÚ THÍCH

1. Việc định kiểu chặt chẽ chặn các thao tác giữa giá trị dữ liệu của các kiểu dẫn xuất khác nhau hoặc giữa kiểu dẫn xuất và kiểu cha, ngoại trừ sử dụng việc chuyển đổi kiểu rõ ràng.

2. Tập các giá trị dữ liệu hoặc thao tác thích hợp của kiểu dẫn xuất có thể rút gọn hoặc mở rộng.

3. Ngược lại với kiểu cha.

15.04.29

quá trình chuyển đổi kiểu

Phép biến đổi cách biểu diễn giá trị dữ liệu của một kiểu dữ liệu sang giá trị dữ liệu của một kiểu dữ liệu khác, thường được thực hiện để tránh một kiểu dữ liệu vi phạm không tương xứng.

CHÚ THÍCH - Quá trình chuyển đổi kiểu giữa các kiểu số thường được phép nhưng có thể làm giảm độ chính xác.

15.04.30

định kiểu chặt chẽ

Việc tuân thủ yêu cầu rằng các toán hạng trong cấu trúc ngôn ngữ phải là kiểu dữ liệu tương thích với kiểu dữ liệu trong thao tác đó hoặc đã chuyển đổi kiểu rõ ràng trước khi thực hiện thao tác đó.

VÍ DỤ: định kiểu chặt chẽ trong Ada coi phép cộng 2 + 3.5 là vi phạm, vì 2 là số nguyên và 3.5 là số thực.

15.04.31

định kiểu yếu

Việc giảm nhẹ các nguyên tắc của định kiểu chặt chẽ.

VÍ DỤ: Định kiểu yếu có thể cho phép phép cộng số nguyên với số dấu phẩy động không cần chuyển đổi kiểu rõ ràng một trong hai toán

creating new data types.

NOTE - Contrast with base type, derived type.

15.04.28

derived type

A data type whose data values and operations are replicas of those of an existing parent type.

NOTES

1. Strong typing prohibits operations among data values of different derived types, or between a derived type and a parent type, unless explicit type conversion is used.

2. The set of data values or the applicable operations of derived types may be reduced or expanded.

3. Contrast with parent type.

15.04.29

type conversion

Transformation of the representation of a data value of one data type to that of another data type, usually performed to avoid an illegal data type mismatch.

NOTE - Type conversion among numeric types frequently is permitted but may cause loss of accuracy, precision, or both.

15.04.30

strong typing

Enforcement of the requirement that operands in a language construct must be of data types compatible with those of the operation or have explicitly undergone type conversion before the operation is performed.

Example: The strong typing in Ada makes the addition of 2 + 3.5 illegal, because 2 is an integer and 3.5 a real number.

15.04.31

weak typing

A relaxation of the rules of strong typing.

Example: Weak typing may permit the addition of an integer and a floating-point number without explicit type conversion of one of the two operands.

hạng đó.

CHÚ THÍCH - Trong định kiểu yếu, có thể có hoặc không có quá trình chuyển đổi kiểu hàm ẩn.

15.04.32

kiểu định trước

Kiểu dữ liệu được tham chiếu bởi một định danh định trước mà ngôn ngữ lập trình cung cấp các thao tác thích hợp đối với nó.

15.04.33

kiểu phổ biến

Kiểu dữ liệu ký tự số và kiểu dữ liệu là kết quả của một vài phép tính định trước tuân thủ theo định kiểu chặt chẽ.

VÍ DỤ: Trong Ada, một khai báo số (không có một kiểu dữ liệu) là một kiểu phổ biến.

15.04.34

ẩn danh

Gắn liền với một đối tượng dữ liệu không có *khai báo* kiểu dữ liệu rõ ràng.

15.04.35

định dạng (trong ngôn ngữ lập trình)

Cấu trúc ngôn ngữ quy định cách biểu diễn theo dạng ký tự, của các đối tượng dữ liệu trong một bản ghi, tệp, thông điệp, thiết bị lưu trữ hoặc kênh truyền.

15.04.36

hình (trong ngôn ngữ lập trình)

Cấu trúc ngôn ngữ để mô tả định dạng *đối tượng dữ liệu* kiểu chuỗi bằng một ký tự in mẫu.

15.05 Câu lệnh và biểu thức

15.05.01

câu lệnh

Đơn vị cú pháp kết thúc hoặc việc biểu diễn khai báo hoặc việc quy định đơn vị công tác bao gồm việc định danh các hoạt động được thực hiện, toán hạng (nếu có) được sử dụng trong việc thực hiện các hoạt động này và việc xếp đặt bất kỳ kết quả nào.

NOTE - In weak typing, there may or may not be implicit type conversion.

15.04.32

predefined type

A data type, referenced by a predefined identifier, for which a programming language provides appropriate operations.

15.04.33

universal type

A data type of numeric literals and the data type of the result of some predefined operations used for compliance with strong typing.

Example: In Ada, a number declaration (without a data type) assumes a universal type.

15.04.34

anonymous

Pertaining to a data object that has no explicit data type *declaration.

15.04.35

format (in programming languages)

A language construct that specifies the representation, in character form, of data objects in a record, file, message, storage device, or transmission channel.

15.04.36

picture (in programming languages)

A language construct that describes the format of string-type data objects by means of a model character literal.

15.05 Statements and expressions

15.05.01

statement

An explicitly terminated syntactic unit either representing a declaration or prescribing a unit of work that includes identification of actions to be performed, operands (if any) to be used in performing these actions, and disposition of any results.

CHÚ THÍCH - Một số ngôn ngữ lập trình không xem khai báo là câu lệnh.

15.05.02

câu lệnh đơn

câu lệnh cơ bản (không sử dụng)

Câu lệnh không kèm theo câu lệnh khác.

15.05.03

câu lệnh ghép

Câu lệnh gồm một hoặc nhiều câu lệnh, được phân định tương đương về cú pháp với câu lệnh đơn.

15.05.04

câu lệnh gán

gán

Câu lệnh đơn thay thế giá trị dữ liệu hiện tại của biến bằng giá trị dữ liệu mới do một biểu thức quy định.

15.05.05

câu lệnh thoát

Câu lệnh đơn kết thúc việc thi hành một cấu trúc ngôn ngữ kèm theo.

15.05.06

câu lệnh trả về

Cấu trúc ngôn ngữ trong mô đun chỉ rõ kết thúc của trình tự thi hành (hoặc có thể vài trình tự như vậy) trong mô đun đó và nhảy tới một điểm cụ thể trong mô đun gọi và cung cấp một kết quả cho mô đun gọi.

15.05.07

trả về (nội động từ)

Để thi hành một câu lệnh trả về mà câu lệnh này gây ra một bước nhảy tới *chương trình gọi*.

15.05.08

trả về (ngoại động từ)

Để cung cấp một giá trị dữ liệu cho *chương trình gọi* khi thi hành một câu lệnh trả về.

NOTE - Some programming languages do not consider declarations to be statements.

15.05.02

simple statement

elementary statement (deprecated)

A statement that encloses no other statement.

15.05.03

compound statement

A statement that contains one or more statements, so delimited as to be the syntactic equivalent of a simple statement.

15.05.04

assignment statement

assignment

A simple statement that replaces the current data value of a variable with a new data value specified by an expression.

15.05.05

exit statement

A simple statement used to end the execution of an enclosing language construct.

15.05.06

return statement

A language construct within a module that designates the end of an execution sequence (or possibly several such sequences) in that module and causes a jump to a specified point in the calling module, and possibly provides a result to it.

15.05.07

to return (intransitive)

To execute a return statement that causes a jump to the calling program.

15.05.08

to return (transitive)

To provide a data value to the calling program when executing a return statement.

15.05.09

mục nhập

Việc khởi tạo một trình tự thi hành khi bắt đầu một chương trình con hoặc tại nơi được chỉ rõ bởi tên mục nhập trong chương trình con đó.

15.05.10

tên mục nhập

Từ định danh định rõ điểm bắt đầu trình tự thi hành.

15.05.11

câu lệnh chuyển tới

Câu lệnh đơn quy định việc truyền câu lệnh điều khiển chương trình từ vị trí của nó trong trình tự thi hành tới câu lệnh đích thường được xác định bởi một nhãn.

CHÚ THÍCH - Việc truyền câu lệnh điều khiển chương trình có thể tương đương với một bước nhảy.

15.05.12

câu lệnh vô điều kiện

câu lệnh bắt buộc

Câu lệnh được thi hành mà không có bất kỳ điều kiện nào.

15.05.13

câu lệnh điều kiện

Câu lệnh ghép để lựa chọn thi hành một trình tự các câu lệnh kèm theo hoặc không thi hành trình tự các câu lệnh đó, phụ thuộc vào giá trị của biểu thức điều kiện tương ứng là một hoặc nhiều điều kiện tương ứng.

VÍ DỤ: Trong Pascal, câu lệnh if và câu lệnh case là câu lệnh điều kiện.

15.05.14

biểu thức điều kiện

Biểu thức mà ước lượng của nó được sử dụng để lựa chọn các trình tự thi hành tiếp theo

15.05.15

câu lệnh if

Câu lệnh điều kiện thi hành trình tự các câu lệnh kèm theo hoặc bỏ qua chúng, phụ thuộc vào giá

15.05.09

entry

The initiation of an execution sequence at the beginning of a subprogram or elsewhere as designated by an entry name in the subprogram.

15.05.10

entry name

An identifier that designates the beginning of an execution sequence.

15.05.11

goto statement

A simple statement that specifies an explicit transfer of program control from its place in the execution sequence to a target statement that usually is identified by a label.

NOTE - The transfer of program control may be equivalent to a jump.

15.05.12

unconditional statement

imperative statement

A statement that is executed without any condition.

15.05.13

conditional statement

A compound statement that selects for execution one or none of the enclosed sequences of statements depending on the value of a conditional expression of one or more corresponding conditions.

Examples: In Pascal, if statements and case statements are conditional statements.

15.05.14

conditional expression

An expression whose evaluation is used to select subsequent execution sequences.

15.05.15

if statement

A conditional statement that causes execution of the enclosed sequences of statements or skips

trị đúng của biểu thức điều kiện.

15.05.16

câu lệnh case

Câu lệnh điều kiện để lựa chọn thi hành một trong số các trình tự các câu lệnh lựa chọn, phụ thuộc vào giá trị của biểu thức điều kiện.

15.05.17

câu lệnh lặp

câu lệnh vòng lặp

Câu lệnh ghép bao gồm cơ chế để điều khiển việc thi hành lặp lại các câu lệnh kèm theo nó.

15.05.18

cấu trúc while

Cấu trúc ngôn ngữ cho điều khiển lặp để xác định một phép chạy thử được thực hiện trước mỗi bước lặp lại.

15.05.19

cấu trúc until

Cấu trúc ngôn ngữ cho điều khiển lặp để xác định phép chạy thử được thực hiện sau mỗi bước lặp lại.

15.05.20

cấu trúc for

Cấu trúc ngôn ngữ cho điều khiển lặp để xác định phép chạy thử thực hiện kiểm soát như vậy, thường dựa trên cơ sở biến điều khiển vòng lặp và quy định thay đổi biến điều khiển lặp được thực hiện giữa các bước lặp.

15.05.21

câu lệnh do while

câu lệnh repeat while

câu lệnh perform while

Câu lệnh lặp lại trong đó điều khiển lặp được kết hợp trong cấu trúc while.

them depending on the truth value of the conditional expression.

15.05.16

case statement

A conditional statement that selects for execution one of a number of alternative sequences of statements depending on the value of a conditional expression.

15.05.17

iteration statement

loop statement

A compound statement that includes a mechanism to control repeated execution of its enclosed statements.

15.05.18

while-construct

A language construct for iteration control that defines a test to be performed before each iteration step.

15.05.19

until-construct

A language construct for iteration control that defines a test to be performed after each iteration step.

15.05.20

for-construct

A language construct for iteration control that defines the test to be performed for such control, usually based on a loop- control variable, and the prescription for the changes of that iteration control variable to be carried out between iteration steps.

15.05.21

do while statement

repeat while statement

perform while statement

An iteration statement where the iteration control is incorporated in a while-construct.

15.05.22

câu lệnh until

câu lệnh repeat until

câu lệnh perform until

Câu lệnh lặp lại trong đó kiểm soát lặp được kết hợp trong cấu trúc until.

15.05.23

câu lệnh perform

Câu lệnh ghép để quy định việc truyền lệnh điều khiển vào một hoặc nhiều thủ tục trong COBOL và quy định kết quả trả về việc thi hành thủ tục được quy định, bất cứ khi nào thủ tục đó hoàn thành.

CHÚ THÍCH - Câu lệnh perform cũng được sử dụng để điều khiển việc thi hành một hoặc nhiều câu lệnh vô điều kiện trong phạm vi của nó.

15.05.24

câu lệnh khối

Mọi trình tự câu lệnh được giới hạn có thể coi như một đơn vị cú pháp đơn và có thể có một định danh.

VÍ DỤ: Trong chương trình Pascal có thể coi như một như một tiêu đề cụ thể theo sau bởi một câu lệnh khối với thủ tục đã xác định tương tự.

CHÚ THÍCH

1. Câu lệnh khối là một thành phần cú pháp cơ bản của ngôn ngữ cấu trúc dạng khối.

2. Trong một vài ngôn ngữ lập trình (ví dụ C++), "khối" đồng nghĩa với câu lệnh ghép. Trong ngôn ngữ lập trình khác (ví dụ Ada) khái niệm này có nghĩa rất cụ thể và có thể gồm các khai báo và các bộ điều khiển ngoại lệ.

3. Việc thực thi câu lệnh khối thường tác động lên phạm vi và thời gian tồn tại (của) các đối tượng dữ liệu được khai báo như một phần của câu lệnh khối.

15.05.25

câu lệnh gọi thủ tục

gọi thủ tục

Câu lệnh đơn cung cấp các *tham số thực* và gọi việc thi hành thủ tục.

15.05.22

until statement

repeat until statement

perform until statement

An iteration statement where the iteration control is incorporated in an until-construct.

15.05.23

perform statement

A compound statement that explicitly specifies transfer of control to one or more COBOL procedures and the return of control implicitly whenever execution of the specified procedure is completed.

NOTE - The perform statement is also used to control execution of one or more unconditional statements which are within its scope.

15.05.24

block statement

Any bounded sequence of statements that can be taken as a single syntactic unit and which may have an identifier.

Example: A Pascal program can be considered simply as a specific header followed by a block statement, with a procedure similarly defined.

NOTES

1. A block statement is a basic syntactic component of block-structured languages.

2. In some programming languages (e.g., C++), "block" is synonymous with compound statement. In other programming languages (e.g., Ada) this concept is given a very specific meaning and may include declarations and exception handlers.

3. Implementation of a block statement usually has an impact on the scope and lifetime of data objects declared as part of a block statement.

15.05.25

procedure-call statement

procedure call

A simple statement that provides the *actual parameters* for and invokes the execution of a procedure.

CHÚ THÍCH - Ngược lại với gọi hàm.

15.05.26

câu lệnh gọi - mục nhập

Câu lệnh đơn cho phép một tác vụ yêu cầu quy định đáp ứng một tác vụ khác.

15.05.27

câu lệnh trễ

Câu lệnh đơn được sử dụng để trì hoãn việc thi hành một tác vụ chứa một yêu cầu trễ.

15.05.28

câu lệnh bỏ qua

Câu lệnh đơn làm cho một hoặc nhiều tác vụ trở nên bất thường và ngăn chặn bất kỳ quy định thêm với tác vụ như vậy.

15.05.29

câu lệnh gia tăng

Câu lệnh đơn truyền một ngoại lệ hoặc khiến ngoại lệ xuất hiện.

15.05.30

câu lệnh chấp nhận

Câu lệnh ghép trong tác vụ máy chủ khiến tác vụ máy chủ này đợi một tác vụ khác hoặc đợi chương trình chính thi hành một câu lệnh gọi mục nhập để đồng bộ hóa tác vụ.

15.05.31

câu lệnh lựa chọn

Câu lệnh ghép cho phép một tác vụ đang gọi hoặc một tác vụ đã gọi lựa chọn các hướng hành động thay thế hoặc đợi.

15.05.32

câu lệnh lựa chọn đợi

Câu lệnh lựa chọn để đợi cuộc gọi từ câu lệnh gọi mục nhập trước khi nó thi hành trình tự câu lệnh của nó.

15.05.33

biểu thức

Cấu trúc ngôn ngữ xác định việc tính toán một giá trị dữ liệu như một kết quả từ một hoặc nhiều toán

NOTE - Contrast with function call.

15.05.26

entry-call statement

A simple statement that permits a task to request a rendezvous with another task.

15.05.27

delay statement

A simple statement used to suspend execution of a task that contains a request for a delay.

15.05.28

abort statement

A simple statement that causes one or more tasks to become abnormal, preventing any further rendezvous with such a task.

15.05.29

raise statement

A simple statement that propagates an exception or causes it to occur.

15.05.30

accept statement

A compound statement within a server task, that causes this server task to wait for another task or for the main program to execute an entry-call statement for task synchronization.

15.05.31

select statement

A compound statement that allows a calling task or a called task to choose alternative courses of action or to wait.

15.05.32

selective-wait statement

A select statement that waits for a call from an entry-call statement before it executes its sequence of statements.

15.05.33

expression

A language construct that defines the computation of a data value as a result from one or more

hạng.

CHÚ THÍCH - Các toán hạng có thể là ký tự, định danh, gọi hàm.

15.05.34

chế độ hỗn hợp (tính từ)

kiểu hỗn hợp (tính từ)

Gắn liền với một biểu thức gồm hai hoặc nhiều kiểu dữ liệu khác nhau.

15.05.35

biểu thức Boolean

Cấu trúc ngôn ngữ xác định việc tính toán giá trị logic.

15.05.36

thứ tự toán tử

Quy tắc sắp xếp xác định trình tự áp dụng các toán tử trong một biểu thức.

CHÚ THÍCH - Quy tắc thứ tự có thể quy định hướng đánh giá.

15.06 Phần chương trình

15.06.01

mô đun

đơn vị chương trình

Một phần của chương trình được xây dựng riêng biệt hoặc đồng nhất tương ứng với các hoạt động như biên dịch, kết nối hoặc thi hành và có thể tương tác với các chương trình hoặc các phần của chương trình khác.

CHÚ THÍCH

1. Khái niệm thuật ngữ "mô đun" có thể thay đổi theo các ngôn ngữ lập trình khác nhau.

2. Xem Hình 2.

15.06.02

Thân chương trình (trong ngôn ngữ lập trình)

Cấu trúc ngôn ngữ bao gồm phần thi hành của một câu lệnh hoặc mô đun.

15.06.03

chương trình con

Mô đun có định danh và được gọi vào luồng điều

operands.

NOTE - Operands may be literals, identifiers, function calls.

15.05.34

mixed mode (adj.)

mixed type (adj.)

Pertaining to an expression that contains two or more different data types.

15.05.35

Boolean expression

A language construct that defines the computation of a logical value.

15.05.36

operator precedence

An ordering rule defining the sequence of the application of operators within an expression.

NOTE - The ordering rule may specify the evaluation direction.

15.06 Parts of programs

15.06.01

module

program unit

A part of a program developed to be discrete or identifiable with respect to actions such as compilation, binding, or execution, and that may interact with other programs or parts of programs.

NOTES

1. The concept referred to by the term "module" may vary according to the different programming languages.

2. See figure 2.

15.06.02

body (in programming languages)

A language construct that comprises the executable part of a statement or module.

15.06.03

subprogram

A module that has an identifier and that is invoked

khiển từ một chương trình khác hoặc bởi một mô đun khác do một cấu trúc ngôn ngữ cụ thể từ đó luồng điều khiển trả về chương trình hoặc mô đun gọi đó.

15.06.04

thường trình chung

Chương trình con, khi được gọi lại sau một lệnh thi hành, thì bắt đầu lại tại vị trí mà lệnh thi hành trước đó đã trả về.

15.06.05

cuộc gọi (trong ngôn ngữ lập trình)

Lệnh để truyền điều khiển từ một mô đun tới một mô đun khác, thường hàm ý rằng kiểm soát quay trở lại mô đun gọi.

CHÚ THÍCH - Cuộc gọi thường quy định các tham số được truyền qua và xuất phát từ *mô đun được gọi*.

15.06.06

gọi (động từ)

Thực hiện một cuộc gọi.

15.06.07

gọi theo tên

Cuộc gọi mà *mô đun gọi* cung cấp cho *mô đun được gọi* tên của một hoặc nhiều tham số để ước lượng thời gian kết hợp tham số sử dụng trong *mô đun được gọi*.

15.06.08

gọi theo tham chiếu

gọi theo địa chỉ

gọi theo vị trí

Cuộc gọi mà *mô đun gọi* cung cấp cho *mô đun được gọi* các địa chỉ của các tham số được truyền qua.

CHÚ THÍCH - Trong cuộc gọi theo tham chiếu, *mô đun được gọi* có khả năng thay đổi các giá trị tham số được lưu trữ bởi mô đun đang gọi.

15.06.09

gọi theo giá trị

Cuộc gọi mà *mô đun gọi* cung cấp tới *mô đun*

into the control flow from another program or by another module by means of a specific language construct and from which the control flow returns to the invoking program or module.

15.06.04

coroutine

A subprogram that, when called again after an execution, resumes at the location to which its previous execution returned.

15.06.05

call (in programming languages)

The instruction to transfer control from one module to another, usually with the implication that control will be given back to the calling module.

NOTE - A call usually specifies parameters to be passed to and from the called module.

15.06.06

to call

To execute a call.

15.06.07

call by name

A call in which the *calling module* provides to the called module the names of one or more parameters to be evaluated each time the associated parameter is used in the called module.

15.06.08

call by reference

call by address

call by location

A call in which the calling module provides to the called module the addresses of the parameters to be passed.

NOTE - In a call by reference, the called module has the ability to change the values of the parameters stored by the calling module.

15.06.09

call by value

A call in which the *calling module* provides to the

được gọi các giá trị thực của tham số được truyền qua.

CHÚ THÍCH - Trong một cuộc gọi theo giá trị, *mô đun được gọi* không thể thay đổi các giá trị các tham số lưu trữ bởi hoặc cho mô đun đang gọi.

15.06.10

gọi chương trình con

cuộc gọi gọi một chương trình con.

VÍ DỤ: Câu lệnh gọi thủ tục, gọi hàm.

15.06.11

thủ tục

thường trình con

Chương trình con không trả về một giá trị dữ liệu ngoại trừ phần cơ cấu tham số.

CHÚ THÍCH

1. Trong COBOL, thủ tục là một đoạn hoặc một nhóm đoạn liên tiếp có lô gic hoặc một phần đoạn (bao gồm 0 hoặc nhiều đoạn) theo phép phân chia thủ tục.

2. Trong một vài ngôn ngữ lập trình (ví dụ C và C++), cấu trúc ngôn ngữ thủ tục không khác với cấu trúc ngôn ngữ chức năng ngoại trừ các giá trị dữ liệu trả về có thể bỏ trống hoặc không sử dụng.

15.06.12

Hàm (trong ngôn ngữ lập trình)

Chương trình con, với các *tham số hình thức* để tạo ra một giá trị dữ liệu để trả về vị trí được gọi.

CHÚ THÍCH - Hàm cũng có thể tạo ra các thay đổi khác bằng việc sử dụng các tham số.

15.06.13

gọi hàm

Cấu trúc ngôn ngữ cung cấp các *tham số thực* để gọi lệnh thi hành một hàm và gây ra lệnh thi hành đó.

CHÚ THÍCH

1. Gọi hàm có thể được sử dụng như một toán hạng trong biểu thức hoặc một *tham số thực* của cuộc gọi chương trình con.

called module the actual values of the parameters to be passed.

NOTE - In a call by value, the called module cannot change the values of the parameters stored by or for the calling module.

15.06.10

subprogram call

A call that invokes a subprogram.

Examples: Procedure-call statement, function call.

15.06.11

procedure

subroutine

A subprogram that does not return a data value, except as part of the parameter mechanism.

NOTES

1. In COBOL, a procedure is a paragraph, or a group of logically successive paragraphs, or a section (consisting of zero or more paragraphs) within the procedure division.

2. In some programming languages (i.e., C and C++), the procedure language construct is not differentiated from the function language construct except that returned data values may be void or not used.

15.06.12

function (in programming languages)

A subprogram, usually with *formal parameters*, that produces a data value which it returns to the place of the invocation.

NOTE - A function may also produce other changes through the use of parameters.

15.06.13

function call

A language construct that provides the *actual parameters* for the invocation of the execution of a function and causes the execution.

NOTES

1. A function call may be used as an operand in an expression or as an *actual parameter* of a subprogram call.

2. Ngược lại với câu lệnh gọi - thủ tục.

15.06.14

gọi giao dịch

Gọi hàm cho phép tác vụ để yêu cầu một quy định với một tác vụ khác.

15.06.15

đơn vị con

Thân mô đun được biên dịch riêng biệt.

15.06.16

body stub

Dạng thân chương trình chỉ ra phần mô đun có thể thi hành được xác định trong một đơn vị con.

15.06.17

kết nối (trong ngôn ngữ lập trình)

Kỹ thuật cho phép sự tương tác giữa các mô đun, các câu lệnh gọi-thủ tục riêng đối với *các thủ tục không đồng bộ.

15.06.18

liên kết tham số được đặt tên

gán theo tên

Trong cuộc gọi chương trình con, việc đặt tên gọi rõ ràng cho các *tham số hình thức* tương ứng với các *tham số thực* để thiết lập liên kết tham số.

CHÚ THÍCH

1. Trong liên kết tham số được đặt tên, các *tham số thực* có thể nhận bất kỳ thứ tự nào.

2. Ngược lại với liên kết tham số vị trí.

15.06.19

liên kết tham số vị trí

Trong cuộc gọi chương trình con, quan hệ của một *tham số thực* với một *tham số hình thức* trong cùng vị trí theo khai báo của chương trình con.

CHÚ THÍCH - Ngược lại với liên kết tham số được đặt tên.

15.06.20

chế độ tham số hình thức

Đặc điểm cho biết *tham số hình thức* có thể được đánh giá mà không phải thay đổi, có thể đưa ra

2. Contrast with procedure-call statement.

15.06.14

transaction call

A function call that permits a task to request a rendezvous with another task.

15.06.15

subunit

The separately compiled body of a module.

15.06.16

body stub

A form of a body that indicates that the executable part of a module is defined in a subunit.

15.06.17

connection (in programming languages)

A technique that enables interaction among modules, particularly procedure-call statements to asynchronous procedures.

15.06.18

named parameter association

assignment by name

In a subprogram call, the explicit naming of the *formal parameters* corresponding to *actual parameters* to establish parameter association.

NOTES

1. In named parameter association, *actual parameters* can be given in any order.

2. Contrast with positional parameter association.

15.06.19

positional parameter association

In a subprogram call, the correspondence of an *actual parameter* with a *formal parameter* in the same position in the declaration of the subprogram.

NOTE - Contrast with named parameter association.

15.06.20

formal parameter mode

A characteristic that indicates whether a *formal parameter* may be evaluated without changing it,

một giá trị mới hoặc có thể được đánh giá và thay đổi.

15.06.21

chỉ lệnh macro

macro

Chỉ lệnh gọi một định nghĩa macro tại mức ngôn ngữ lập trình đang gọi.

15.06.22

gọi macro

Câu lệnh gọi định nghĩa macro tại mức ngôn ngữ lập trình đang gọi.

15.06.23

định nghĩa macro

Trình tự định trước của các lệnh, câu lệnh hoặc các chỉ lệnh thay thế tương việc gọi chỉ lệnh macro hoặc hàm gọi macro.

15.06.24

gói (trong ngôn ngữ lập trình)

Mô đun được thiết kế để đưa ra cách tóm lược, sự đóng gói hoặc thông tin ẩn thông qua việc nhóm các cấu trúc ngôn ngữ liên quan đến logic, như kiểu dữ liệu và đối tượng dữ liệu của kiểu dữ liệu này, chương trình con với các tham số của kiểu dữ liệu này.

15.06.25

khai báo gói

Việc khai báo riêng của các cấu trúc ngôn ngữ mà các đặc tả của chúng yêu cầu nằm ngoài gói đó cho cả mục đích ghép nối cũng như biên dịch.

15.06.26

phần hiện rõ

Phần khai báo gói cung cấp chi tiết do người sử dụng yêu cầu về các đối tượng hoặc dịch vụ gói.

15.06.27

phần riêng

Phần khai báo gói cung cấp chi tết cấu trúc cần thiết bởi quá trình xây dựng nhưng không liên

may be given a new value, or may be evaluated and changed.

15.06.21

macro instruction

macro

An instruction that invokes a macro definition at the level of the calling programming language.

15.06.22

macro call

A statement that invokes a macro definition at the level of the calling programming language.

15.06.23

macro definition

A predefined sequence of commands, statements, or instructions that replaces each corresponding invoking macro instruction or macrocall.

15.06.24

package (in programming languages)

A module designed to provide abstraction, encapsulation, or information hiding through grouping of logically related language constructs, such as data types, data objects of these data types, and subprograms with parameters of these data types.

15.06.25

package declaration

The separate declaration of those language constructs whose specifications are required outside the package, either for interfacing or for compilation purposes.

15.06.26

visible part

That part of a package declaration that provides details required by users of objects or services of the package.

15.06.27

private part

That part of a package declaration that provides structural details needed by the development

quan và không thể truy cập đối với người dùng chức năng của gói đó.

15.06.28

chung

Gắn liền với một cấu trúc ngôn ngữ sử dụng như một khuôn mẫu để tạo một cấu trúc ngôn ngữ thực để đối với các kiểu dữ liệu có thể áp dụng tuân theo các quy tắc định kiểu chặt chẽ.

15.06.29

khai báo chung

Khai báo một cấu trúc ngôn ngữ chung để đưa ra các tham số chung được thay thế bởi các *tham số thực* trong một thể hiện chung.

15.06.30

Thân chương trình chung

Thân của một ngôn ngữ cấu trúc chung sử dụng như một khuôn mẫu cho các thân của các cấu trúc ngôn ngữ thực tương ứng trong một thể hiện chung.

15.06.31

thao tác chung

Thao tác bị quá tải và không chỉ rõ một thao tác cụ thể nhưng đưa ra các *tham số hình thức* cho các *tham số thực* của các kiểu dữ liệu cụ thể.

VÍ DỤ: Đơn vị từ vựng "+" có thể có nghĩa là: phép cộng số nguyên, phép cộng số thực, phép hội tập hợp, kết nối thông tin, v.v...

15.06.32

gói chung

Gói được thiết kế để đưa ra các khuôn mẫu cho các thuật toán hoặc thao tác liên quan.

VÍ DỤ: Các gói chung cho các hàm lượng giác, thao tác ngăn xếp, hàm trong tài chính, v.v...

15.06.33

mô đun chung

Khuôn mẫu được tham số hóa để tạo các mô đun bởi thể hiện chung.

process but irrelevant and unaccessible to the functional users of the package.

15.06.28

generic

Pertaining to a language construct that serves as a template for creating an actual language construct for applicable data types in compliance with the rules of strong typing.

15.06.29

generic declaration

The declaration of a generic language construct that introduces the generic parameters which are to be replaced by *actual parameters* during a generic instantiation.

15.06.30

generic body

The body of a generic language construct that serves as a template for the bodies of corresponding actual language constructs during a generic instantiation.

15.06.31

generic operation

An operation that is overloaded and does not designate one specific operation but rather provides *formal parameters* for *actual parameters* of specific data types.

Example: The lexical token "+" can mean integer addition, real addition, set union, concatenation, etc.

15.06.32

generic package

A package designed to provide templates for related algorithms or operations.

Examples: Generic packages for trigonometric functions, stack operations, financial functions, etc.

15.06.33

generic module

A parameterized template for creating modules by generic instantiation.

CHÚ THÍCH - Các tham số của khuôn mẫu có một bản chất chung và không nhầm lẫn với các *tham số hình thức* của các mô đun kết quả.

15.06.34

thể hiện chung

Quá trình phân tích các tham số chung từ một mô đun chung để tạo ra một mô đun cụ thể.

15.06.35

trường hợp chung

Mô đun cụ thể được tạo từ một mô đun chung bởi thể hiện chung.

15.07 Tác vụ

15.07.01

chương trình chính

Mô đun đầu tiên của một chương trình được thi hành và có thể gọi lệnh thi hành của các mô đun khác.

15.07.02

tác vụ (trong ngôn ngữ lập trình)

Mô đun có thể thi hành đồng thời với các mô đun khác trên bộ đa xử lý hoặc thi hành lần lượt trong một bộ xử lý đơn.

CHÚ THÍCH - Phân biệt giữa tác vụ với mô đun về quan điểm kiểm soát lệnh thi hành thường không chính xác.

15.07.03

phần tới hạn

Một phần tác vụ trong khoảng thời gian thi hành tác vụ đó các phần khác của tác vụ này hoặc các tác vụ khác bị cấm thi hành.

15.07.04

đồng bộ hóa tác vụ

Biện pháp mà nhờ đó các tác vụ phối hợp các hoạt động của chúng cùng một lúc.

VÍ DỤ: Cờ hiệu, màn hình, điểm quy định.

NOTE - The parameters of the template are of a generic nature and should not be confused with *formal parameters* of the resulting modules.

15.06.34

generic instantiation

The process of resolving generic parameters from a generic module in order to create a concrete module.

15.06.35

generic instance

A concrete module created from a generic module by generic instantiation.

15.07 Tasks

15.07.01

main program

The first module of a program to be executed and that may invoke the execution of other modules.

15.07.02

task (in programming languages)

A module that can be executed concurrently with other modules either on a multiprocessor or interleaved on one processor.

NOTE - The distinction between a task and a module from the point of view of execution control is not always precise.

15.07.03

critical section

A portion of a task during the execution of which other parts of this or other tasks are prohibited from execution.

15.07.04

task synchronization

The means by which tasks coordinate their activities in time.

Examples: Semaphore, monitor, rendezvous.

15.07.05

điểm quy định

Việc tương tác giữa hai tác vụ được sắp xếp theo thời gian tại một điểm nào đó trong mỗi quá trình thi hành tác vụ và tại đó một quá trình đợi quá trình khác.

15.07.06

cờ hiệu

Cấu trúc dữ liệu để kiểm soát bởi một hàng đợi, truy cập tới tài nguyên sẵn có cho nhiều tác vụ, nhưng tại một thời điểm chỉ cho một tác vụ.

15.07.07

màn hình (trong ngôn ngữ lập trình)

Đối tượng dữ liệu chia xẻ cùng với tập các thao tác có thể vận dụng đối tượng dữ liệu đó để kiểm soát các yêu cầu về tài nguyên hoặc truy cập vào các tài nguyên sẵn có cho các quá trình song song, nhưng tại một thời điểm chỉ cho một quá trình.

15.08 Thi hành

15.08.01

trình tự thi hành

Thứ tự các khai báo và của lệnh thi hành các câu lệnh và các phần của câu lệnh.

15.08.02

luồng kiểm soát

Đường dẫn trình tự thi hành có thể chiếm qua một chương trình

CHÚ THÍCH - Khái niệm trừu tượng của tất cả các luồng kiểm soát có thể được biểu diễn bởi sơ đồ luồng kiểm soát.

15.08.03

hiệu ứng phụ

Mọi hệ quả gián tiếp gây ra bởi việc thi hành một biểu thức, câu lệnh hoặc chương trình phụ.

CHÚ THÍCH - Hiệu ứng cạnh có thể được dự đoán, ví dụ, để thay đổi giá trị dữ liệu của một tham số được

15.07.05

rendezvous

The interaction between two tasks which is time-coordinated at a certain point in each process of task execution and where one process may wait for the other.

15.07.06

semaphore

A data structure for controlling, by means of a queue, access to the resources that are available to more than one task, but only to one task at a time.

15.07.07

monitor (in programming languages)

A shared data object together with a set of operations that may manipulate the data object in order to control requests for resources or access to the resources that are available to parallel processes, but only to one process at a time.

15.08 Execution

15.08.01

execution sequence

The order of the elaboration of declarations and of the execution of statements and parts of statements.

15.08.02

control flow

A path the execution sequence may take through a program.

NOTE - An abstraction of all the control flows can be represented by a control-flow diagram.

15.08.03

side effect

Any indirect consequence caused by the execution of an expression, statement, or subprogram.

NOTE - Side effects may be intended, for example, to change the data value of a parameter passed by a

truyền bởi một hàm.

15.09 Lập trình hướng đối tượng

15.09.01

ẩn thông tin

Nguyên tắc từ chối truy cập tới hoặc kiến thức về một cấu trúc ngôn ngữ hoặc các chi tiết cụ thể, ngoại trừ các chi tiết được cho là thiết yếu đối với người sử dụng để hiểu.

15.09.02

tóm lược (động từ)

Áp dụng việc ẩn thông tin đối với một cấu trúc ngôn ngữ.

15.09.03

việc tóm lược

Quá trình hoặc kết quả tóm lược.

15.09.04

riêng

Gắn với các đặc điểm của cấu trúc ngôn ngữ mà không trực tiếp sẵn có đối với người sử dụng các cấu trúc ngôn ngữ này.

15.09.05

đối tượng (trong ngôn ngữ lập trình)

Tập các thao tác và dữ liệu lưu trữ và duy trì tác động của các thao tác đó.

CHÚ THÍCH - Các đối tượng được thực thi như các các gói hoặc tác vụ trong Ada, như "mô đun" trong Modula-2 và như "các đối tượng" trong Smalltalk.

15.09.06

thông điệp (trong ngôn ngữ lập trình)

Yêu cầu đối với một đối tượng để thực hiện một trong các thao tác của nó.

15.09.07

giao thức (trong ngôn ngữ lập trình)

Tập hợp các quy tắc xác định cách hoạt động của đối tượng trong trao đổi thông điệp.

15.09.08

phương pháp (trong ngôn ngữ lập trình)

Thao tác mà đối tượng thi hành dựa trên việc xác

function.

15.09 Object-oriented programming

15.09.01

information hiding

The principle of denying access to or knowledge of a language construct or specific details there of, except for those details considered essential for the user to know.

15.09.02

to encapsulate

To apply information hiding to a language construct.

15.09.03

encapsulation

The process or the result of encapsulating.

15.09.04

private

Pertaining to characteristics of language constructs that are not directly available to the user of these language constructs.

15.09.05

object (in programming languages)

A set of operations and data that store and retain the effect of the operations.

NOTE - Objects are implemented as packages or tasks in Ada, as "modules" in Modula-2, and as "objects" in Smalltalk.

15.09.06

message (in programming languages)

A request for an object to perform one of its operations.

15.09.07

protocol (in programming languages)

The set of rules that determines the behavior of objects in the exchange of messages.

15.09.08

method (in programming languages)

An operation that an object executes upon receipt

nhận của thông điệp.

15.09.09

lớp (trong ngôn ngữ lập trình)

Khuôn mẫu đối với các đối tượng định nghĩa cấu trúc bên trong và tập các thao tác đối với các trường hợp cụ thể của đối tượng như vậy.

CHÚ THÍCH - Trong lập trình hướng đối tượng, lớp có thể so sánh với kiểu dữ liệu trong một vài ngôn ngữ lập trình như C và Pascal.

15.09.10

tính đa dạng

Khả năng đáp ứng khác nhau với cùng một thông điệp của các đối tượng khác nhau.

15.09.11

kế thừa

Việc sao chép toàn bộ hoặc một phần cấu trúc bên trong và việc sao chép tập các thao tác từ một lớp đến một lớp cấp dưới.

15.09.12

ủy quyền

Phương pháp cho phép một đối tượng gán việc sử dụng thông điệp cho một đối tượng khác.

15.09.13

hướng đối tượng

Gắn với một kỹ thuật hoặc ngôn ngữ lập trình hỗ trợ các đối tượng, lớp và kế thừa.

CHÚ THÍCH - Một số tổ chức có thẩm quyền liệt kê các yêu cầu đối với lập trình hướng đối tượng như sau: ẩn thông tin hoặc đóng gói, trìu tượng hóa dữ liệu, truyền thông điệp, tính đa dạng, ràng buộc động và kế thừa.

15.10 Tính năng và đặc điểm

15.10.01

lập chỉ số dưới dòng

Việc tham chiếu một phần tử mảng bởi một tham chiếu mảng và một hoặc nhiều biểu thức. Khi được đánh giá, biểu thị vị trí của phần tử đó.

of a message.

15.09.09

class (in programming languages)

A template for objects that defines the internal structure and the set of operations for instances of such objects.

NOTE - In object-oriented programming, classes are comparable to data types in some programming languages, such as C and Pascal.

15.09.10

polymorphism

The ability of different objects to respond to the same message differently.

15.09.11

inheritance

The copying of all or part of the internal structure and of the set of operations from one class to a subordinate class.

15.09.12

delegation

A means that permits an object to assign servicing of a message to another object.

15.09.13

object-oriented

Pertaining to a technique or a programming language that supports objects, classes, and inheritance.

NOTE - Some authorities list the following requirements for object- oriented programming: information hiding or encapsulation, data abstraction, message passing, polymorphism, dynamic binding, and inheritance.

15.10 Features and characteristics

15.10.01

subscripting

Referencing an array element by means of an array reference and one or more expressions that, when evaluated, denote the position of the element.

15.10.02

tham chiếu gián tiếp

Việc tham chiếu qua một đối tượng dữ liệu để trỏ tới cấu trúc ngôn ngữ được tham chiếu.

CHÚ THÍCH - Tham chiếu có thể tiến hành dọc theo một dãy các đối tượng dữ liệu, trong trường hợp đó, mỗi đối tượng dữ liệu, ngoại trừ đối tượng dữ liệu cuối cùng, trỏ tới đối tượng tiếp theo, đối tượng dữ liệu cuối cùng trỏ tới cấu trúc ngôn ngữ được tham chiếu.

15.10.03

khởi tạo (động từ)

Mang một giá trị dữ liệu đến một đối tượng dữ liệu khi bắt đầu thời gian sống.

15.10.04

cấp phát bộ nhớ động

cấp phát không gian lưu trữ cho các đối tượng dữ liệu chỉ trong khoảng thời gian thực thi trong phạm vi của chúng.

15.10.05

khả năng mở rộng

Khả năng của một ngôn ngữ lập trình cho phép đặc tả các cấu trúc ngôn ngữ mới và sử dụng chúng giống về cú pháp với tiêu chuẩn các cấu trúc ngôn ngữ.

15.10.02

indirect referencing

Referencing via a data object that points to a referenced language construct.

NOTE - The referencing may be done along a chain of data objects, in which case each data object, except the last, points to the next, the last data object pointing to the referenced language construct.

15.10.03

to initialize

To give a data value to a data object at the beginning of its lifetime.

15.10.04

dynamic storage allocation

Allocation of storage space to data objects only for the duration of the execution of their scope.

15.10.05

extensibility

The capability of a programming language to allow the specification of new language constructs and their use in the same syntactic manner as the standard language constructs

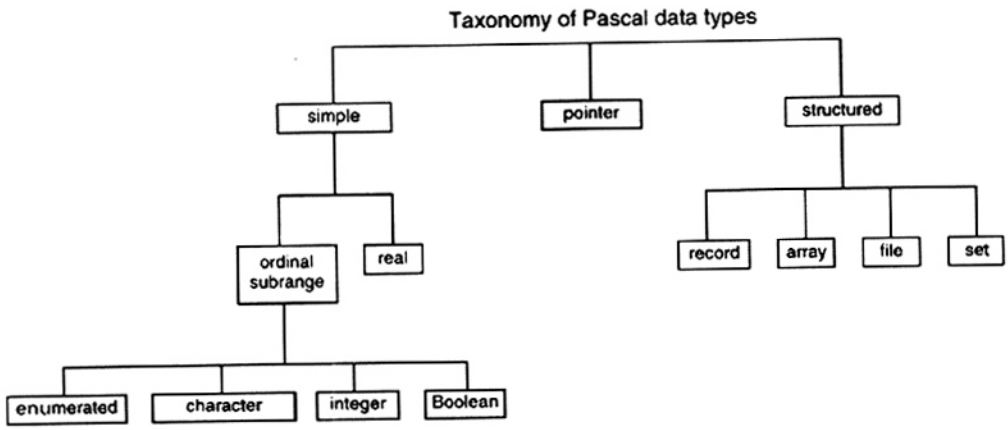
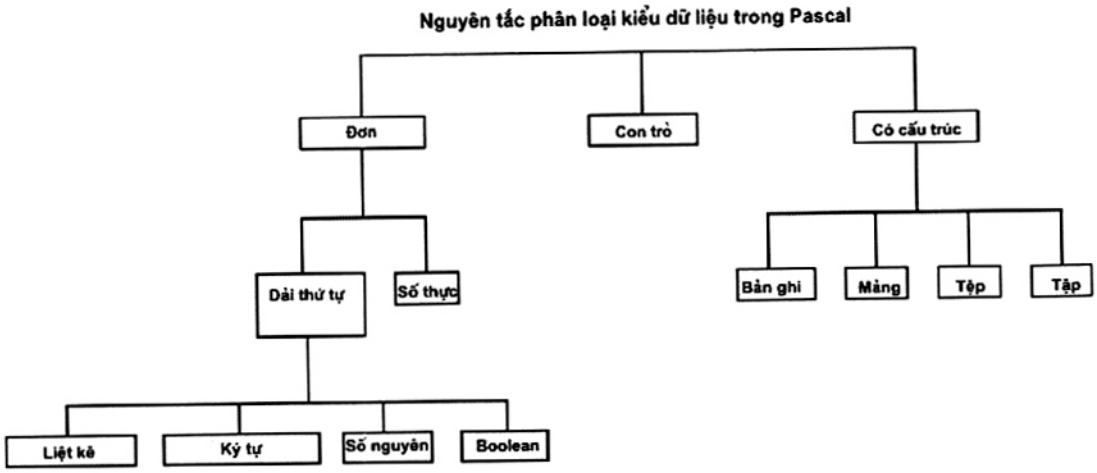


Figure 1 - Examples of data types in Ada and Pascal



Hình 1 - Ví dụ kiểu dữ liệu trong Ada và Pascal

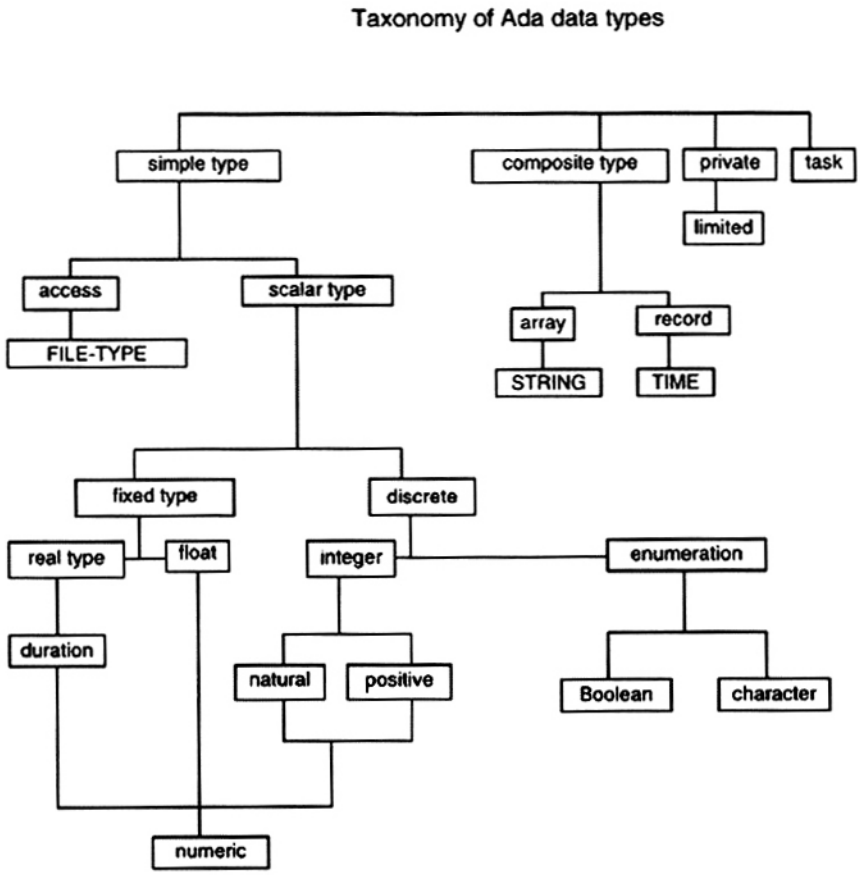
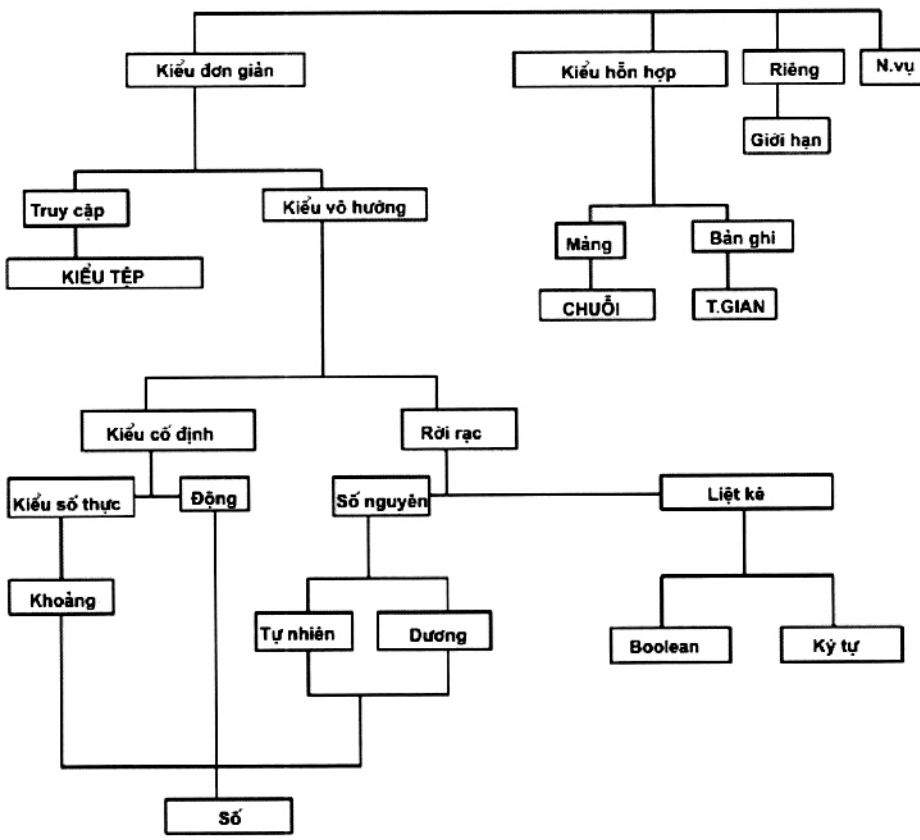


Figure 1 - Examples of data types (*cont.*)

Legend : CAPITAL LETTERS indicate predefined identifiers.
All lower case indicates reserved words in ADA

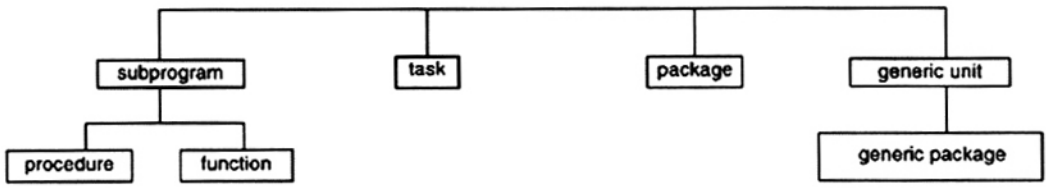
Nguyên tắc phân loại kiểu dữ liệu trong Ada



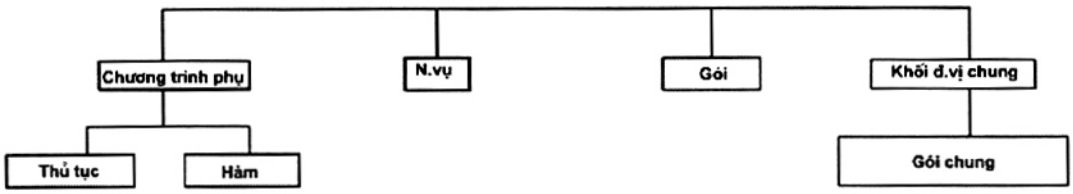
Hình 1- Ví dụ kiểu dữ liệu (tiếp)

Chú giải : CHỮ HOA chỉ các định danh định trước.
Tất cả chữ thường chỉ các từ dành riêng trong ADA

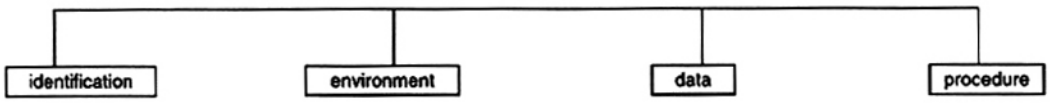
Ada program units (modules)



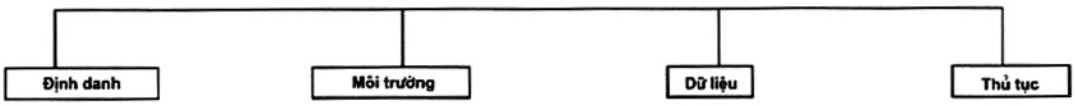
Các khối (môđun) chương trình trong Ada



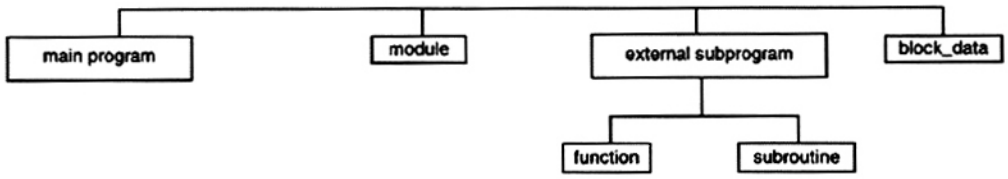
COBOL divisions



Các phân khu trong COBOL



Fortran program units (units of compilation)



Các khối chương trình trong Fortran (khối biên dịch)