CONTENTS

CHAPTER – 8

ABSTRACT DATA TYPE

A DATA TYPE

In programming languages a ***data type*** is an attribute of a piece of data that tells the computer (and the programmer) something about what kind of data is being dealt with. This involves setting constraints on the data, such as what values that data can take on, and what operations may be performed on that data. Common data types may include: integers, floating-point numbers (decimals), and alphanumeric strings. For example, in the Java programming language, the "int" type represents the set of 32-bit integers ranging in value from -2,147,483,648 to 2,147,483,647, as well as the operations that can be performed on integers, such as addition, subtraction, and multiplication. Colors, on the other hand, are represented by three bytes denoting the amounts each of red, green, and blue, and one string representing that color's name; allowable operations include addition and subtraction, but not multiplication.

In a broad sense, a **data type** defines a set of values, and the allowable operations on those values. Almost all programming languages explicitly include the notion of **data type**, though different languages may use different terminology. Most programming languages also allow the programmer to define additional data types, usually by combining multiple elements of other types and defining the valid operations of the new data **type**.

A **data type** can also be thought of as a constraint placed upon the interpretation of data in a **type system**, describing representation, interpretation and structure of ***values*** or ***objects*** stored in computer memory. The ***type system*** uses **data type** information to check correctness of computer programs that access or manipulate the data.

MACHINE DATA TYPE

All data in computers based on digital electronics is represented as bits (alternatives 0 and 1) on the lowest level. The smallest addressable unit of data is a group of bits called a **byte** (usually an octet, which is 8 bits). The unit processed by machine code instructions is called a word (as of 2006, typically 32 or 64 bits). Most instructions interpret the word as a binary number, such that a 32-bit word can represent unsigned integer values from 0 to 232 − 1 or signed integer values from − 231 to 231 − 1. Because of two's complement, the machine language and machine don't need to distinguish between these unsigned and signed data types for the most part. There is a specific set of arithmetic instructions that use a different interpretation of the bits in word as a floating-point number.

PRIMITIVE DATA TYPES

In computer science, ***primitive types*** are provided by a programming language as basic building blocks. **Primitive types** are also known as built-in types or basic types.

Depending on the language and its implementation, **primitive types** may or may not have a one-to-one correspondence with objects in the computer's memory. However, one usually expects operations on **primitive types** to be the fastest language constructs there are. Integer addition, for example, can be performed as a single machine instruction, and some processors offer specific instructions to process sequences of characters with a single instruction. In particular, the C standard mentions that "a 'plain' int object has the natural size suggested by the architecture of the execution environment". Values of the **primitive types** do not share state. Most languages do not allow the behavior or capabilities of **primitive types** to be modified by programs. Exceptions include Smalltalk, which permits **primitive data types** to be extended within a program, adding to the operations that can be performed on them or even redefining the built-in operations.