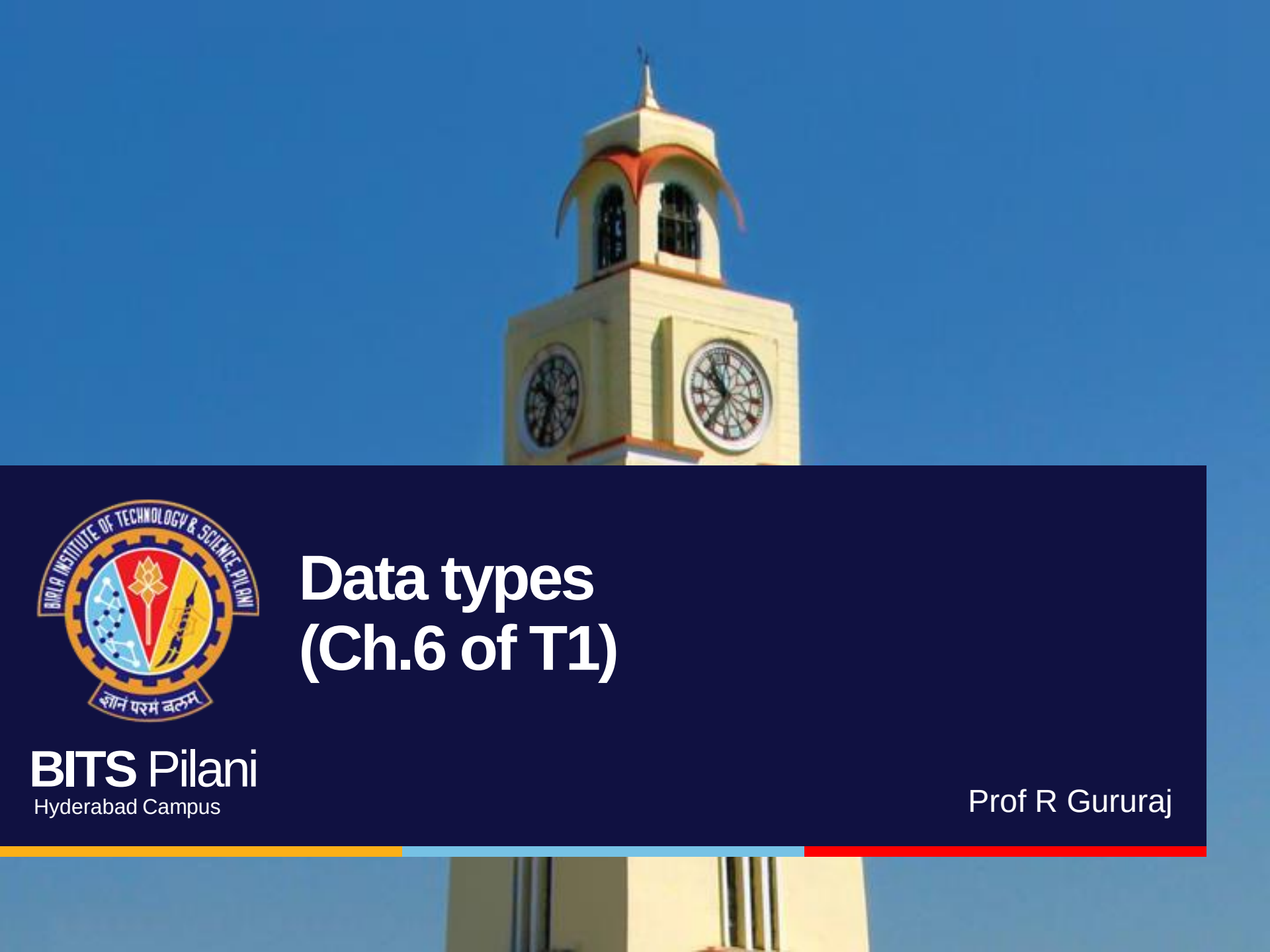
Principles of Programming Languages(CS F301)

BITS Pilani
Hyderabad Campus

Prof.R.Gururaj
CS&IS Dept.



Data types (Ch.6 of T1)

BITS Pilani
Hyderabad Campus

Prof R Gururaj

Introduction

A *data type* defines a collection of data values and set of operations on those values.

Best approach introduced by ALGOL-60, was to provide a few basic types and a few flexible structure defining operators that allow a programmer to design a data structure as per the need.

This was one of the most important advancement in the evolution of programing languages.

User defined data types:

More readable and easy to modify.

Next step in evolution was *abstract data* types, supported by most of the programming languages designed after 1980.

The fundamental idea is that the interface of a type, which is visible to the user is separated from the representation and set of operations on values of that type which are hidden from the user.

All of the types provided by High-level languages are abstract data types.

Most common structured data types in imperative languages are arrays and records.

Primitive data types

Numeric types:

Integer: byte , short , int , long in Java

Floating point: real numbers.

On many computers, floating point numbers are stored in binary.

One issue with floating point is loss of accuracy.

Even 0.1 can't be represented by finite number of digits.

Most newer machines use IEEE754 format.

Language implementers use whatever scheme supported by the HW.

Many languages support float and double.

The double type is provided for situations where larger fractional parts / larger range of exponents is needed.

Precision: accuracy of fractional part. Number of bits.

Range: combination of range of fractional part and exponents.

Decimal type: Fixed number of decimal points useful in domains like finance/business.

Boolean types: Java and C++ pure Boolean type.

C# supports use of numerical values in place.

Character string types:

values consist of sequence of characters.

Design issues:

- should be special kind of char arrays or primitive type.
- length static or dynamic.

String operations.

Java supports StringBuffer.

C# and Ruby also have String class.

Python includes string primitive.

Perl, Java Script, Ruby etc., string matching using regular expressions. Class libraries of C#, Java, Python, C# support pattern matching.

String length.

Java strings fixed, immutable. C and C++ use arrays.

Implementing Character Strings:

Could be directly supported by HW.

Most common is SW implementation for String storage , retrieval and manipulation.

Use Descriptor for referencing static character strings.

Contains name, length, and address of the first character.

Dynamic string's descriptors need max length and current length.

Use linked lists but it is complex but simple.

Array of pointers pointing to heap storage for a each character.

User defined data types

An ordinal type is one in which the range of possible values can be easily associated with set of positive integers.

In Java integer, char and Boolean are predefined ordinal types.

There are two user-defined ordinal types that are supported by PL.

Enumeration.

Subrange.

Enumeration type: is one in which all of the possible values which are named constants are enumerated are provided in the definition.

enum days = { Mon, Tue, Wed, Thu, Fri, Sat, Sun}

The enumeration constants are typically implicitly assigned the integer values 0, 1, … (implicit)

Design issues:

1,. Can the constants appear in more tan one definition?

2. Are enumeration values coerced to integer.

3. Are any other type coerced to emumeration type.

Subrange type: is a contiguous subsequence of an ordinal type.

Ex: 12..14 is a subrange of integer type introduced by Pascal and included in Ada.

Ada:

type days is = { Mon, Tue, Wed, Thu, Fri, Sat, Sun};

subtype weekdays is days range Mon..Fri;

subtype index is Integer range 1..100;

Sec 5.0 Array type: is a homogenous aggregate of data elements in which an individual element is identified in the aggregate relative to the first element.

The individual elements are of same type.

In C, C++, Java , Ada, C# they store same types.

In Python, Ruby, Java Script the array elements are pointers / ref to other objects.

Array Design issues:

What types are legal for subscripts.

Are subscripting expressions range checked.

When are subscript ranges bound.

When does array allocation take place.

Can allocation and initialization happen together.

What kind of slices are allowed.

Note: we will cover more details in Tuts.

Associative Arrays:

An associative array is an unordered collection of data elements that are indexed by an equal number of values called keys.

In case of an non-associative array the indices are never stored because of their regularity.

In an associative array the user defined key must be stored.

It is a key value pair.

Supported directly by: Python , Ruby, Perl etc..

Supported by class libraries in Java, C++, C# and F#.

F# (pronounced **F sharp; 2005**) is a functional-first, general purpose, strongly typed, multi-paradigm programming language that encompasses functional, imperative, and object-oriented programming methods.

Perl associative arrays also called as *hashes*.

%salaries = (“Tom” => 57000, “Ram” => 80000, “Sam” => 60000);

reassigning values.

%salaries{“Tom”}=90000;

delete %salaries{“Sam”};

@salaries=(); //emptied

$ val= %salaries{“Ram”};

It is dynamic.

7.Record types

A record is an aggregate of data items in which individual elements are identified by name and accessed through offset from the First element.

Design issues.

What is the syntactic form of references to fields.

Are elliptical references allowed.

COBOL Record type definition

```
01   EMPLOYEE-RECORD.
     02   EMPLOYEE-NAME.
          05   FIRST     PICTURE IS X(20).
          05   MIDDLE    PICTURE IS X(10).
          05   LAST      PICTURE IS X(20).
     02   HOURLY-RATE  PICTURE IS 99V99.
```

Ada example for record.

```
type Employee_Name_Type is record
   First : String (1..20);
   Middle : String (1..10);
   Last : String (1..20);
end record;
type Employee_Record_Type is record
   Employee_Name: Employee_Name_Type;
   Hourly_Rate: Float;
end record;
Employee_Record: Employee_Record_Type;
```

8.Tuple type:

A tuple is a data type that is similar to record, except that the elements are not named.

Ex Python:

```
myTuple  = (3, 5.8, ‘Apple’)
myTuple[1]
```

9.List types:

First supported by LISP.

Scheme.

(A B C D)

Data and code have same syntax.

If (A B C) is code it is for calling function A with parameters B and C.

10.Union types:

A union is a type whose variables may store different type values at different times during program execution.

Design issues:

- Should type checking be required? Note that any such type checking be dynamic.
- Should unions be embedded in records?

Free Unions: Ex: C

No type checking while processing.

Discriminated unions (Ex: ALGOL68, Ada Haskell etc)

10.Pointer types:

A pointer type is one in which the variables have a range of values that consist of memory addresses and a special value NIL.

Uses:

Provide same power of indirect addressing.

Provide way to manage dynamic storage.

Design Issues

The primary design issues particular to pointers are the following:

- What are the scope and lifetime of a pointer variable?
- What is the lifetime of a heap-dynamic variable (the value a pointer references)?
- Are pointers restricted as to the type of value to which they can point?
- Are pointers used for dynamic storage management, indirect addressing, or both?
- Should the language support pointer types, reference types, or both?

Dangling Pointer

Lost dynamic-heap variable

Memory leakage

Ada pointer is called access type.

12. Type checking

It is an activity of ensuring that the operands of an operator are of compatible types.

A compatible type is one that either is legal for the operator or is allowed under language rules to be implicitly converted by compiler to legal type.

This automatic conversion is called coercion.

In Java, int→ float.

Type checking at run time is called dynamic type checking.

(Java Script, and PHP)

Type checking is complicated for unions etc.

Strong Typing:

A language is strongly typed if type errors are always detected.

Ada is nearly strongly typed. Because the programmer can by pass type checking if needed.

C and C++ not strongly typed because of UNION types.

ML and F# are strongly typed.

Java and C# are also same as Ada.

Summary of Ch.6

innovate achieve lead

1. Introduction
2. Primitive types
3. Char Strings
4. User defined ordinal types
5. Array types
6. Associative arrays
7. Record/ Tuple/ List types
8. Union type
9. Pointer type
10. Type checking