3.5a UNION

Objectives

- To study the syntax and usage of union with examples
- To study how it differs from Structure

Agenda

- Union (definition in detail)
- Syntax
- Example with its memory allocation
- Example to show how to use union members
- Differences between union and structures

Union

- A union is a user defined data type like structure.
- The union groups logically related variables into a single unit.
- The union data type allocate the space equal to space need to hold the largest data member of union.
- The union allows different types of variable to share same space in memory. (i.e. a single variable may hold different types at different times)
- Members are overlaid on top of each other.
- It is *programmer's responsibility* to keep track of which type is stored in a **`union`** at any given time!
- There is no other difference between structure and union than internal difference.
- The method to declare, use and access the union is same as structure.

Syntax

- A union has to defined, before it can used. The syntax of defining a structure is

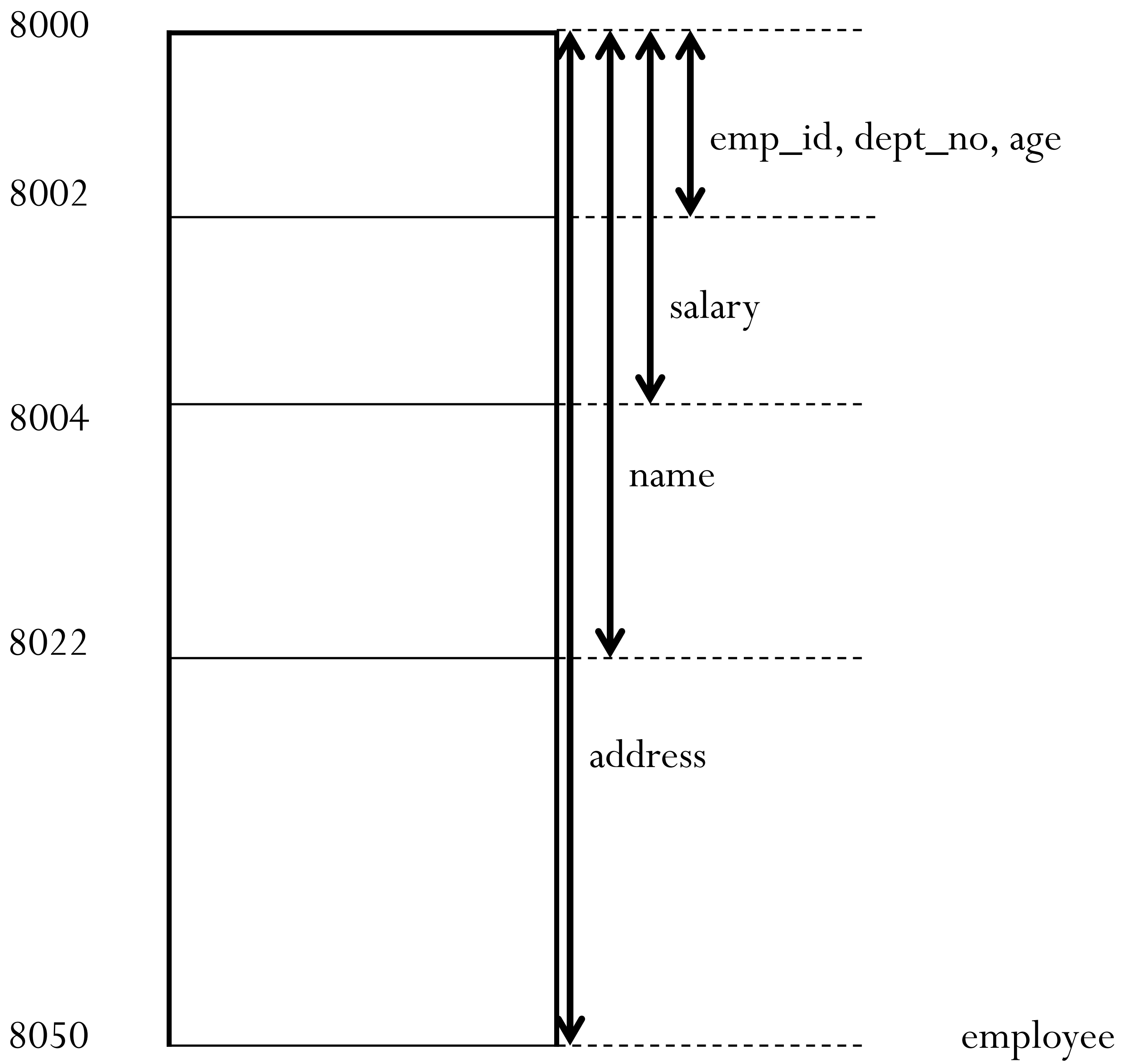
```
 union <union_name>
 {
  <data_type> <variable_name>;
  <data_type> <variable_name>;
   ……..
  <data_type> <variable_name>;
 };
```

Example

- The union of Employee is declared as

```
union employee
{
        int  emp_id;
        char name[20];
        float salary;
        char address[50];
        int dept_no;
        int age;
};
```

Memory Space Allocation



Example

- Up to programmer to determine how to interpret a union (i.e. which member to access)
- Often used in conjunction with a "type" variable that indicates how to interpret the union value

```
union VALUE {
  float f;
  int i;
  char *s;
};

enum TYPE { INT, FLOAT, STRING };
struct VARIABLE {
  enum TYPE type;
  union VALUE value;
};
```

Access `type` to determine how to interpret `value`

Unions (continued)

- Union may only be initialized to a value appropriate for the type of its first member
- **`unions`** are used much less frequently than **`structures`** — mostly used
  - in the inner details of operating system
  - in device drivers
  - in embedded systems where you have to access registers defined by the hardware

Difference between Structures & Union

1. The memory occupied by structure variable is the sum of sizes of all the members but memory occupied by union variable is equal to space hold by the largest data member of a union.

2. In the structure all the members are accessed at any point of time but in union only one of union member can be accessed at any given time.

Summary

- Discussed the syntax of Union with an example.
- Discussed the how to use the Union members with an example.
- Discussed the differences between the Structure and Union.