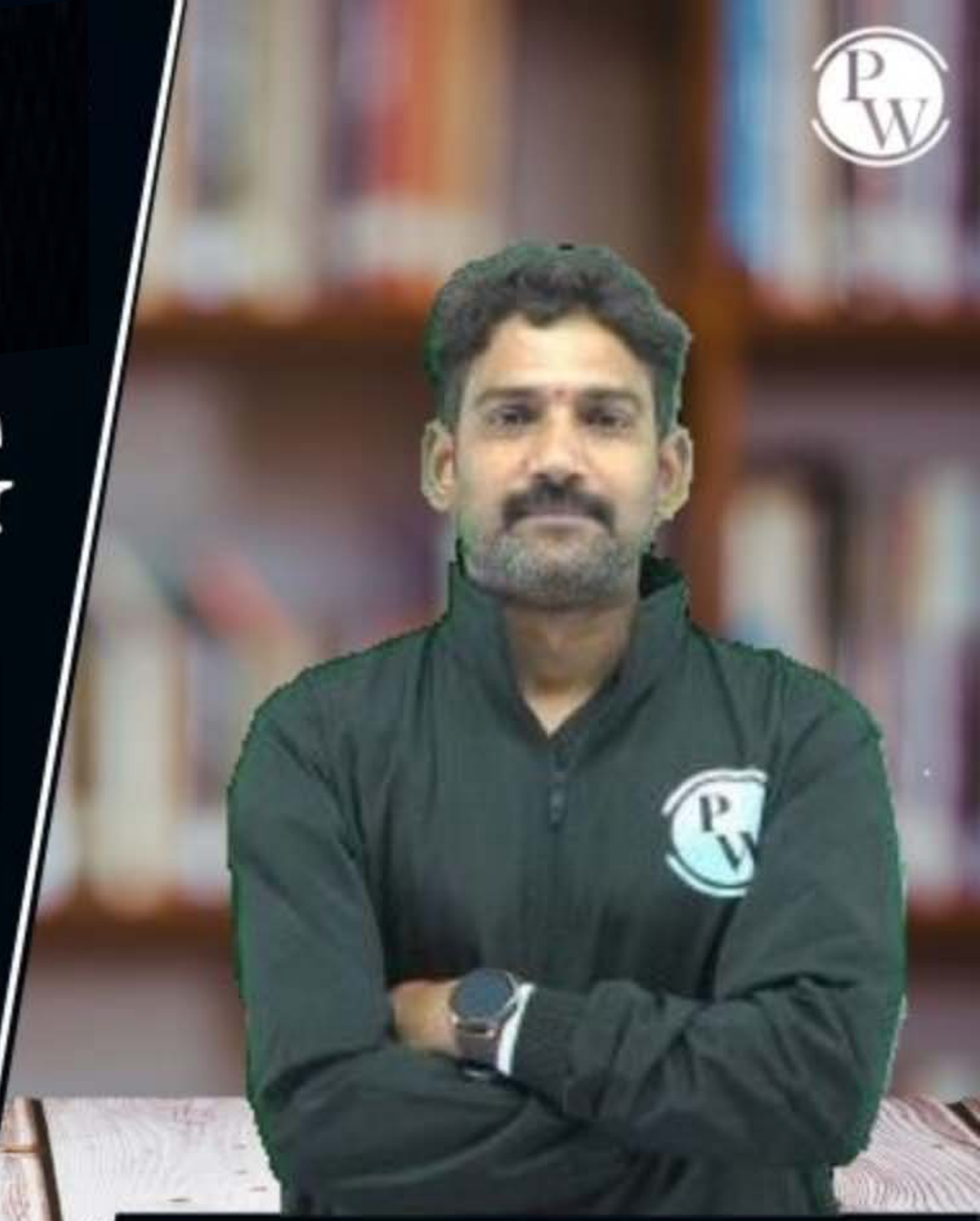
CS & IT ENGINEERING

[redacted]

'C' Programming

Structures & Unions

Lecture No.- 03



By- Satya sir

Recap of Previous Lecture

- Nested Structures & Unions
  - Size of Variables

Topics to be Covered

- Nested Structures & unions
- Assign values to Members
- Structure vs union
- typedef keyword
- Self-referential Structure
- PYQ Practice

Topic : Structures & Unions - 3

Let 1 int = 4 Bytes

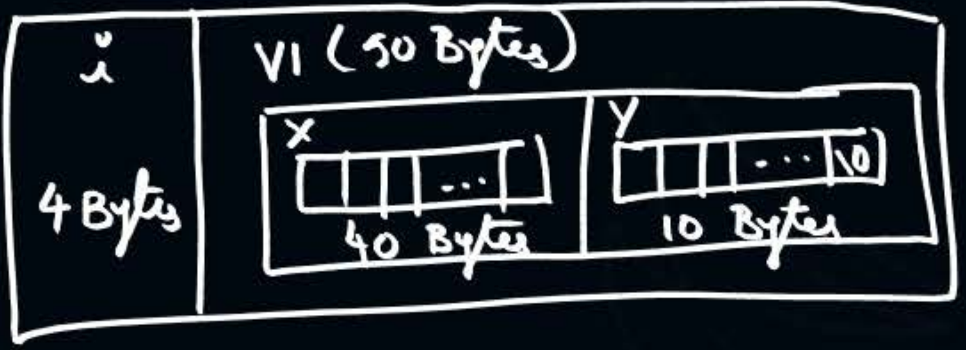
Nested Structures & Unions

Ex:

```
struct xyz
{
    int x[10];     → 10*4=40B
    char y[10];    → 10*1=10B
                   ─────────────
};                 Σ = 50Bytes

struct ABC
{
    int i;          → 4Bytes
    struct xyz V1;  → 50Bytes
                    ─────────────
} V2;               Σ = 54 Bytes
```

V2 (54 Bytes)



Topic : Structures & Unions - 3

Assign values to Members

- The values are assigned in the order of declaration.

- Default values will be
  - zero (integers)
  - '\0' (characters)
  - 0.0 (Real numbers)

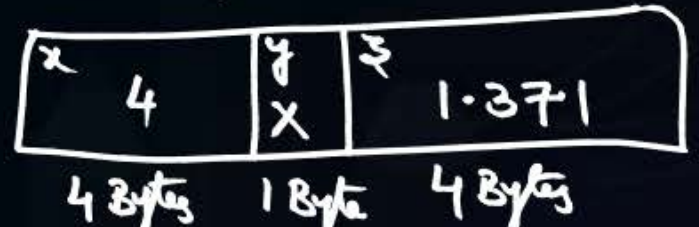
Ex:

1 int = 4Bytes

```
struct ABC
{
    int x;
    char y;
    float z;
};
```

```
void main( )
{
  struct ABC V1 = { 4, 'x', 1.371 };
  Printf("%d %c %f", V1.x, V1.y, V1.z);
}
```

V1 (9 Bytes) Contiguous



Topic : Structures & Unions - 3

Random assignment to Members

Syntax : Struct Name Variable = { .member = Value . . . };

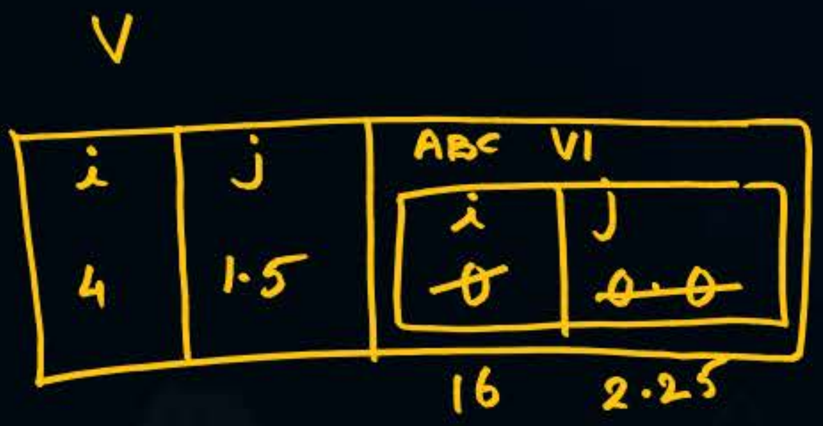
Ex:
```
struct ABC
{
    int x ;
    float y ;
    char z ;
};
void main( )
{
    struct ABC V1 = {.z = 'X', .x = 4, .y = 1.317};
}
```

Topic : Structures & Unions - 3

Examples

①
```
struct ABC
{
    int i;
    float j;
};
Struct XYZ
{
    int i;
    float j;
    Struct ABC V1;
};
```

```
void main(    )
{
    Struct XYZ V = { 4, 1.5 };
    V.V1.i = V.i * V.i;
    V.V1.j = V.j * V.j;
    Printf(" %d, %f", V.V1.i, V.V1.j);  // 16, 2.25
}
```



Topic : Structures & Unions - 3

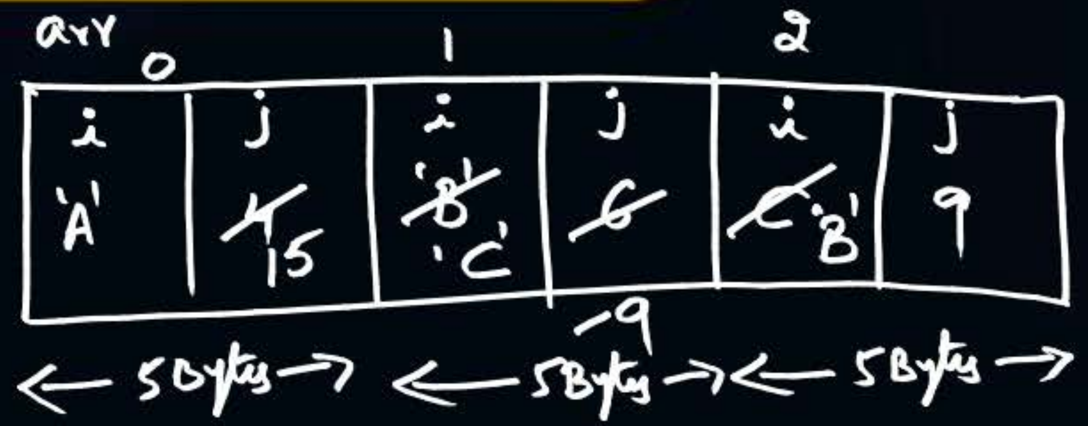
Ex: 2

1 int = 4 Bytes

```
struct Array
{
    char i;
    int j;
};

void main( )
{   int i;
    Struct Array arr[3] = {{'A', 4}, {'B', 6}, {'C', 9}};
    arr[0].j = arr[1].j + arr[2].j;      arr[0].j = 6+9 = 15
    arr[1].i++;
    arr[2].i--;
    arr[1].j -= arr[0].j;      arr[1].j = 6-15 = -9
    for(i=0; i<3; i++)
    { Printf("%c, %d \n", arr[i].i, arr[i].j);
    }
}
```

Array of structures



```
o/p:  A  15
      C  -9
      B  9
```

Topic : Structures & Unions - 3

typedef keyword : It is used to assign alias names for Existing datatypes.

Syntax: typedef Existing type New Name;

Ex: 1

```
void main( )
{ typedef int Number;
  typedef char letter;

   Number x = 4 ;
   letter y = 'y';

  Printf(" %d %c", x, y);
}
```

Ex: 2

```
Struct ABC
{ int i;
  float j;
};
typedef Struct ABC ;
Void main( )
{
   ABC V1 = { 4, 1.35 };
}
```

Ex: 3

```
typedef struct ABC
{ int i;
  float j;

};
void main( )
{ ABC V1 = { 4, 1.35 };
}
```

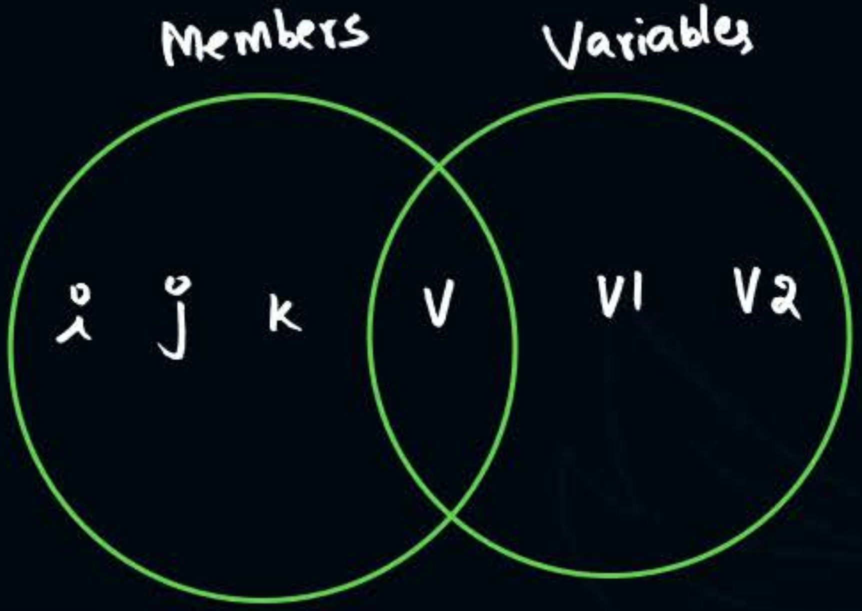
Topic : Structures & Unions - 3

Self-Referential structure [structure only]

- A structure whose member/variable is Variable/member respectively, of itself is known as Self-Referential structure.
- Variables must be Pointer variables to limit self reference, by assigning to NULL.

Ex:

```
struct ABC
{
    int i;
    char j;
    float k;
    struct ABC *V;
}*V1, *V2;
```



- In Linked List implementations, Self-Referential Structure Used for creation of Nodes.

Limitations of Structure / union

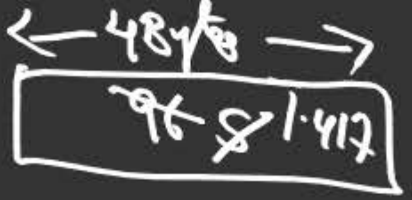
1. members Can not be initialized at the time of Declaration.
2. No storage class Can be applied, default will be automatic
3. functions Cannot be defined Inside structure / union.

Structure Vs Union

Structure	Union
- struct keyword	- union keyword
- Size of Each Variable = Sum of all members Size (Ignoring Padding)	- Size of Each Variable = Max (All Members Size) (Ignoring Padding)
- Simultaneously any / all members Can be accessed.	- At any time, one member only Can be accessed.

```
union ABC
{ int i;     //2 B
  char j;    //1 B
  float k;   // 4 B
} V1;
```

Max = 4B



```
V1.i = 96;
V1.j = 'S';
V1.k = 1.417;
```

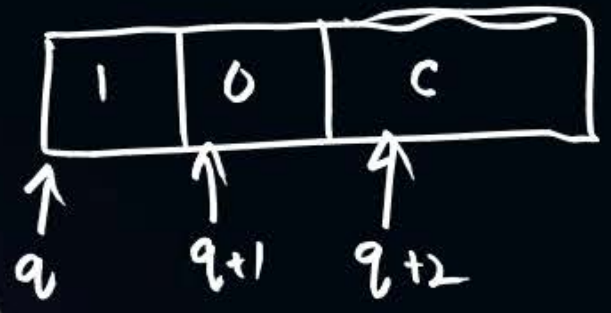
Topic : Structures & Unions - 3

GATE 2018

#Q. Consider the following C program.

```
#include< stdio.h >
struct Ournode{
 char x,y,z;
};
int main(){
 struct Ournode p = {'1', '0', 'a'+2};
 struct Ournode *q = &p;
 printf ("%c, %c", *((char*)q+1), *((char*)q+2));
 return 0;
}
```

Value at   Type casted   Value at



O/P : 0 c

The output of this program is:

a. 0, c ✓
b. 0, a+2
c. ‘0’, ‘a+2’
d. ‘0’, ‘c’

Topic : Structures & Unions - 3

#Q. The following C declarations

GATE 2000

```
struct node{
  int i:
  float j;
};
struct node *s[10];
```

$(*s)[10]$

⇒ Pointer to array of 10 Elements

⇒ Array of $\overset{10}{\text{Pointers}}$

define s to be

A. An array, each element of which is a pointer to a structure of type node ✓
B. A structure of 2 fields, each field being a pointer to an array of 10 elements
C. A structure of 3 fields: an integer, a float, and an array of 10 elements
D. An array, each element of which is a structure of type node

Topic : Structures & Unions - 3

#Q. Consider the following C declaration GATE 2000

```
struct {
    short s[5];      —— 5×2=10B
    union {
    float y;         —— 4B
    long z;          —— 8B
    } u;             Max = 8B
}t;  ————→ 10+8 = 18 Bytes
```

Assume that objects of the type short, float and long occupy 2 bytes, 4 bytes and 8 bytes, respectively. The memory requirement for variable t, ignoring alignment considerations, is

A. 22 Bytes
B. 14 Bytes
C. 18 Bytes ✓
D. 10 Bytes

2 mins Summary

- Structure vs Union
- Assignment of values
- Self referential structure
- Array of structures
- PYQ Practice

THANK - YOU