C LANG TEST-10 (STRUCTURE & UNION)

Total points 50/50

STUDENT NAME *

VIVA

✓ 1. A structure in C is — * 1/1

- A) A collection of variables of the same type
- B) A collection of variables of different types ✓
- C) A function
- D) A keyword

✓ 2. Which keyword is used to define a structure in C? * 1/1

- A) struct ✓
- B) structure
- C) strct
- D) record

✓ 3. What is the correct syntax to declare a structure? * 1/1

- (●) A) struct student { int roll; char name[20]; }; ✓
- ( ) B) structure student (int roll; char name[20];);
- ( ) C) struct student = { int roll; char name[20]; };
- ( ) D) student struct { int roll; char name[20]; };

✓ 4. How do you access members of a structure using a variable? * 1/1

- ( ) A) structure.variable
- (●) B) variable.member ✓
- ( ) C) variable->member
- ( ) D) member.variable

✓ 5. Which operator is used to access structure members through a pointer? *1/1

- ( ) A) . (dot)
- (●) B) -> (arrow) ✓
- ( ) C) *
- ( ) D) &

✓ 6. A structure variable can be declared as — * 1/1

- A) Inside main() only
- B) Globally or locally ✓
- C) Inside functions only
- D) None

✓ 7. The memory size of a structure is — * 1/1

- A) Equal to sum of all members
- B) Equal to size of largest member
- C) Undefined
- D) Depends on compiler alignment ✓

✓ 8. What will this code do? * 1/1

```
struct A {
  int x;
  char y;
};
```

- ○ A) Creates an object
- ◉ B) Defines a new data type ✓
- ○ C) Declares a variable
- ○ D) None

✓ 9. You can create a variable of structure student as — * 1/1

- ○ A) student s;
- ◉ B) struct student s; ✓
- ○ C) structure student s;
- ○ D) record student s;

✓ 10. What is the size of the structure below (on 32-bit system)? * 1/1
struct s { int a; char b; };

- ○ A) 5
- ◉ B) 8 (due to padding) ✓
- ○ C) 4
- ○ D) 6

✓ 11. A structure can contain — * 1/1

- A) Only arrays
- B) Only pointers
- (●) C) Other structures ✓
- D) Only integers

✓ 12. When a structure is nested inside another, it is called — * 1/1

- (●) A) Nested structure ✓
- B) Compound structure
- C) Union
- D) None

✓ 13. Which statement correctly accesses nested structure member? * 1/1

- (●) A) student.address.city ✓
- B) student->city
- C) student(city)
- D) city.student

✓ 14. Structures in C are stored in — * 1/1

- A) Stack
- B) Heap
- (●) C) Memory depending on declaration ✓
- D) ROM

✓ 15. Can structures have arrays as members? * 1/1

- (●) A) Yes ✓
- B) No
- C) Only int arrays
- D) Only char arrays

✓ 16. What does typedef do when used with structure? * 1/1

- A) Defines new variable
- (●) B) Creates alias name for structure ✓
- C) Allocates memory
- D) None

✓ 17. Which is correct typedef usage? * 1/1

- (●) A) typedef struct student { int id; } STUD; ✓
- ( ) B) typedef struct STUD { int id; };
- ( ) C) typedef student struct;
- ( ) D) typedef {int id;} student;

✓ 18. Which of the following initializes a structure correctly? * 1/1

- (●) A) struct student s = {1, "Viva"}; ✓
- ( ) B) struct student s(1, "Viva");
- ( ) C) student s{1, "Viva"};
- ( ) D) new student(1, "Viva");

✓ 19. What is the output? * 1/1

```
struct s { int a; };

struct s x = {10};

printf("%d", x.a);
```

- (●) A) 10 ✓
- ( ) B) 0
- ( ) C) Garbage
- ( ) D) Error

✓ 20. Can structure members be of pointer type? * 1/1

- (●) A) Yes ✓
- ( ) B) No
- ( ) C) Only for int
- ( ) D) Only for char

✓ 21. What is the size of a union? * 1/1

- ( ) A) Sum of sizes of all members
- (●) B) Size of largest member ✓
- ( ) C) Average size
- ( ) D) Undefined

✓ 22. Which keyword defines a union? * 1/1

- ( ) A) struct
- (●) B) union ✓
- ( ) C) record
- ( ) D) type

✓ 23. Which statement is true for union? * 1/1

- (●) A) All members share the same memory ✓
- ( ) B) Each member has its own memory
- ( ) C) It behaves like array
- ( ) D) None

✓ 24. What happens when one member of a union is assigned a value? * 1/1

- (●) A) Only that member is valid ✓
- ( ) B) All members get same value
- ( ) C) All members change
- ( ) D) None

✓ 25. Which is the correct declaration of a union? * 1/1

- ( ) A) union data { int i; float f; };
- ( ) B) struct data (int i; float f;);
- ( ) C) union { int i; float f; } data;
- (●) D) A and C ✓

✓ 26. Which is true for both structure and union? * 1/1

- ○ A) Both can have members of different types
- ○ B) Both use . and -> operators
- ○ C) Both can be passed to functions
- ◉ D) All of the above ✓

✓ 27. What is the output? * 1/1

```
union u { int i; char c; };

union u x;

x.i = 65;

printf("%c", x.c);
```

- ○ A) 65
- ◉ B) A ✓
- ○ C) Error
- ○ D) Garbage

✓ 28. In structure, memory is allocated — * 1/1

- ◉ A) For all members separately ✓
- ○ B) For only one member
- ○ C) Dynamically
- ○ D) Depends on variable

✓ 29. In union, memory is allocated — * 1/1

- ○ A) For all members
- ◉ B) For largest member ✓
- ○ C) For first member only
- ○ D) None

✓ 30.What will be the output? * 1/1

```
union test { int i; float f; };

union test t;

t.i = 10;

printf("%d", t.i);
```

- ◉ A) 10 ✓
- ○ B) Garbage
- ○ C) Error
- ○ D) None

✓ 31. Can a union contain a structure as a member? * 1/1

- (●) A) Yes ✓
- ( ) B) No
- ( ) C) Only if structure is empty
- ( ) D) None

✓ 32. Which is not allowed in structure but allowed in union? * 1/1

- (●) A) Same memory for all members ✓
- ( ) B) Pointer to itself
- ( ) C) Nested types
- ( ) D) None

✓ 33. Which is more memory-efficient? * 1/1

- ( ) A) Structure
- (●) B) Union ✓
- ( ) C) Both equal
- ( ) D) Depends on compiler

✓ 34. Can you assign one structure variable to another of same type? * 1/1

- (●) A) Yes ✓
- ( ) B) No
- ( ) C) Only if pointer
- ( ) D) Only if typedef used

✓ 35.What is printed? * 1/1

```
struct A { int x; float y; };

struct A a1 = {1, 2.5};

struct A a2;

a2 = a1;

printf("%d %.1f", a2.x, a2.y);
```

- ( ) A) 0 0.0
- (●) B) 1 2.5 ✓
- ( ) C) Garbage
- ( ) D) Error

✓ 36. What is the main difference between structure and union? * 1/1

- (●) A) Structure allocates separate memory; union shares memory ✓
- ( ) B) Union allocates separate memory; structure shares
- ( ) C) Both same
- ( ) D) None

✓ 37. Which function can pass structure by reference? * 1/1

- (●) A) Using pointer ✓
- ( ) B) By value
- ( ) C) By both
- ( ) D) None

✓ 38. Structure variables are usually accessed by — * 1/1

- ( ) A) Dot operator
- ( ) B) Arrow operator
- (●) C) Both ✓
- ( ) D) None

✓ 39. Which of the following is invalid? * 1/1

- A) struct student s1, s2;
- B) union data d1, d2;
- (●) C) struct data {int a;} = {10}; ✓
- D) typedef struct data d;

✓ 40. In C, you can use structure to — * 1/1

- A) Group related data
- B) Create user-defined types
- C) Pass multiple data to functions
- (●) D) All of these ✓

✓ 41. Which operator is used to access structure through pointer? * 1/1

- (●) A) -> ✓
- B) .
- C) *
- D) &

✓ 42. Which of the following correctly defines an array of structures? * 1/1

- (●) A) struct student s[10]; ✓
- ( ) B) struct student[10] s;
- ( ) C) student struct[10];
- ( ) D) array student[10];

✓ 43. How do you access the roll number of 3rd student in an array s? * 1/1

- ( ) A) s.roll[2]
- (●) B) s[2].roll ✓
- ( ) C) s->roll[2]
- ( ) D) s.roll->2

✓ 44. What will this print? * 1/1

```
union data { int i; char c; };
union data d;
d.c = 'A';
printf("%d", d.i);
```

- A) 65
- (●) B) Garbage ✓
- C) Error
- D) 0

✓ 45. Which of the following can have bit fields? * 1/1

- A) struct
- B) union
- (●) C) Both ✓
- D) None

✓ 46. A bit field in structure is defined to — * 1/1

- (●) A) Save memory for small data ✓
- B) Increase size
- C) Store characters
- D) Store float

✓ 47. Which of the following can be nested within structure? * 1/1

- ○ A) Another structure
- ○ B) Array
- ○ C) Union
- ◉ D) All ✓

✓ 48. Can a function return a structure? * 1/1

- ◉ A) Yes ✓
- ○ B) No
- ○ C) Only pointer to structure
- ○ D) None

✓ 49. Which of the following statements is false? * 1/1

- ◉ A) Structures can contain functions ✓
- ○ B) Structures can contain arrays
- ○ C) Unions can contain arrays
- ○ D) Unions can store one member at a time

✓ 50. Which of the following is the best use of union? * 1/1

- (●) A) When different data types share the same memory location ✓
- ( ) B) When all members are used simultaneously
- ( ) C) For storing identical data types
- ( ) D) For memory wastage