**1. What is the 8-bit chain of -126 using two's complement?**

The 8-bit chain of -126 using two's complement is **1000010**

Converse 126 into binary form: 01111110 (make sure that there are 8 bits)

Converse it into one's complement: 10000001 (0->1 and 1->0)

Converse it into two's complement 10000010 (plus 1, keep 8 bits)

**2. Based on IEEE-754, write the sequence of bits in the single precision of the number -25.45?**

The sign bit is **1**

The sequence of bits of integral part (23) is **11001** (5 bits)

The sequence of bits of fractional part (.45) is **01110011001100110011001** (23bits)

The binary form of the above number is **11001.01110011001100110011001**

The standardized binary form of the above number is **1.10010111001100110011001\* 2 ^ 4**

The sequence of bits of the exponent part is **10000011** (8 bits)

The sequence of bits of the above number is **11000001110010111001100110011001** (32 bits)

3. **Given the following array declaration in C:**

**int x[10];**

Assume that the size of an int object is 4, and the starting address of the variable x is 1000, what is the address of element x[5]?

The address of element x[5] is **1020**

*The address of element x[5] is 1000 + ((5-0)\* 4) where 0 is the lower bound, 4 is the size of an element.*

**4. Given the following array declaration:**

**int x[4][6][5]; // the lower bound is 0**

Assume that the size of. an int element is 4, the elements of an array are allocated in row-major order, and the starting address of the variable x is 1000, what is the address of the element x[2][4][3]?

The address of the element x[2][4][3] is **1252**

*the address of the element x[2][4][3] is 1000 + (((((2 - 0) \* 6) + (4 - 0)) \* 5) + 3) \* 4 = 1252*

**5**. **Given the following struct declarations in C:**

struct A {

int a;

double b;

float c;

};

struct B {

double a;

float b;

int c;

};

Assume that the size of int, float and double are 4, 4, and 8, respectively. What are the size of struct A and B?

The size of struct A is **24**

The size of struct B is **16**

*There are a 4-byte padding between a and b and another 4-byte padding after c in struct A so the size of struct A is 24. There is no padding in struct B so the size of struct B is 16. Note that two structs have the same number of fields but just different in the order of these field.*

**6. Given the following record declaration in Ada:**

type Shape is (Circle, Triangle, Rectangle);

type Colors is (Red, Green, Blue);

type Figure (Form: Shape) is record

Filled: Boolean;

Color: Colors;

case Form is

when Circle => Diameter: Float;

when Triangle =>

Leftside, Rightside: Integer;

Angle: Float;

when Rectangle => Side1, Side2: Integer;

end case;

end record;

Assume that the size of Boolean, enumeration, Integer, and Float are 1, 2, 2 and 4, respectively. What is the size of an object in type Figure without padding?

The size of an object in type Figure without padding is **13**

*The size of Form is 2 (enumerate)*

*The size of Filled is 1 (Boolean)*

*The size of Color is 2 (enumeration)*

*The size of union is the max size of its components => 8 (Leftsize (2), Rightsize (2) and Angle (4))*

*The total is 13.*

**7. Given the following code fragment in a programming language using fixed-length string:**

x: string[6];

y: string[6];

x = "You";

y = "are";

print(x + y); //+ is string concatenation operator

What is the result of the print statement?

The result of the print statement is **" You are "**

*When x is assigned the value "You", it is also appended 3 more spaces to make its size 6 so x keeps "You ". Similarly, y keeps "are ".*

**8.** **Given the following declaration of a set:**

x: set of 10..73;

Assume that an object in set type is implemented by a bit chain, what is the size of x in byte?

The size of x is **8 bytes**

*There are 64 values from 10 to 73 so an object in this type needs 64 bits = 8 bytes.*

9. Which type requires the initialization must be performed when a variable is declared in this type?

a. Array

b. Struct

c. Pointer

**d. Reference**

**10. Given the following declaration:**

x: array [1..10] of record

a: array [5..10] of integer;

b: record

c: real;

d: array[1..3] of real;

end;

end;

Use type expression to write the type of x? Note that use \* for product.\

The type expression to express the type of x is

**array(1..10,record((a\*array(,5..10,integer)\*(b\*record((c\*real)\*(d\*array(1..3,real)))))))**

11. Given the code fragment as follows:

def foo(x,f) = f(f(x))

What is the type of function foo? Note that if a variable type is used, its name is T.

The type of function foo is **(T1\*(T1->T1)) -> T1**

*1. foo is a function so its type is T1->T2 where T1 is input type and T2 is output type*

*2. This function has 2 parameter, so T1 = T3\*T4 where T3 is the type of x (2a) and T4 is the type of f (2b)*

*3. In the body of the function, there is expression f(x), so f is a function and its type is T5 -> T6 (3a) and x is passed to function f and from (2a) => T5 = T3 (3b)*

*4. There is also expression f(f(x)) and (3a) => T5 = T6 (4)*

*5. The result of expression is also the result of function foo and (1) and (3a) => T2 =T6 (5)*

*6. From (1) and (2a) => type of foo is T3\*T4 -> T2 (6)*

*7. From (6) and (3a) => type of foo is (T3\*(T5->T6)) -> T2 (7)*

*8. From (7) and (3b), (4), (5) => type of foo is (T3\*(T3->T3))->T3*

*There is only one variable type in the result so we can write the type of foo is (T\*(T->T))-> T*