C Complicated Declaration Answers

Sayak Haldar
IIEST, Shibpur

1) Answer is b) Compile time error

Explanation: no identifier is undeclared in main.

2) Answer is c) hello

3) Answer is b) Compile time error.

Explanation: We want to initialize the member no (Initialization: value assignment during declaration) during the time of structure declaration.

More Explanation:

What does the following mean?

```
struct student
{
        int no;
        char name[20];
};
```

We define a identifier student within the struct name space. However, we basically create a type (type declaration) which can be further used to define variables of the new type.

More Explanation:

Name spaces of identifiers

any point in a translation unit, the syntactic context disambiguates uses that refer to different entities.

Thus, there are separate name spaces for various categories of identifiers, as follows:
— label names (disambiguated by the syntax of the label declaration and use);

— the tags of structures, unions, and enumerations (disambiguated by following any32) of the keywords struct, union, or enum);

— the members of structures or unions; each structure or union has a separate name space for its members (disambiguated by the type of the expression used to access the member via the . or -> operator);

— all other identifiers, called ordinary identifiers (declared in ordinary declarators or as enumeration constants).

4) Answer is c) Compile time error

Explanation: The program will cause compilation error due to the following reasons:

- The variable s of newly defined type student should be declared as:

- struct student s: (this will be the change in line 9.
- We cannot assign a string to a character array like the following:
  - s.name = "hello";
  - It should be strcpy(s.name,”hello”)

5) Answer is c) Junk.

Explanation: Since, the variable s of newly defined type student is declared locally, s.name holds junk value (due to no assignment)

6) Answer is a) Nothing

Explanation: Since, the variable s of newly defined type student is declared globally, s.name holds '\0' (without assignment)

7) Answer is a) Compile time error.

Explanation: int *((*x)())[2]; this declaration at line 4 declares a function pointer.

x	--x
*x	--x, a pointer
(*x)()	--x, a function pointer
*((*x)())	--x, a function pointer, points to a function which returns a pointer
int *((*x)())[2]	--x, a function pointer, points to a function which returns a pointer to an integer array of size 2

Now, this does not cause error. But, this is not initialized to a function. So, x() will cause error

Also, int *((*x)())[2] this at line 8, would be something like following:

- int *((y)())[2] where y is the function name of such function which returns a pointer to an integer array

8) Answer is a) y is pointer to the function which returns pointer to an integer array.

Explanation: Just check the answer for the previous question

9) This is not an MCQ question.

int(*f1)(float)

f1	--f1
*f1	--f1, a pointer
(*f1)()	--f1, a function pointer
(*f1)(float)	--f1, a function pointer, which accepts some value of float datatype
int (*f1)(float)	--f1, a function pointer, which accepts some value of float datatype and returns a value of

int datatype

So, f1 is a function pointer which accepts some value of float datatype as only argument to it an returns some value of type int

10)
(*(*f2)(double))(float)

f2	--f2
*f2	--f2, a pointer
(*f2)()	--f2, a function pointer
(*f2)(double)	--f2,a function pointer, points to a function which takes a double datatype argument
*(*f2)(double)	--f2, a function pointer, points to a function which takes a double datatype variable as an argument, returns a function pointer
(*(*f2)(double))()	--f2, a function pointer, points to a function which takes a double datatype variable as an argument, returns a function pointer
(*(*f2)(double))(float)	--f2, a function pointer, points to a function which takes a double datatype variable as an argument, returns a function pointer which could points to function taking a float datatype variable as an argument and returns nothing

11)
int (*(*(*f3)(int))(double))(float)

f3	--f3
*f3	--f3, a pointer
(*f3)()	--f3, a function pointer
*(*f3)(int)	--f3, a function pointer to a function, which points to a function which takes some variable /value of type int as an argument and returns a pointer
((*f3)(int))(double))	-f3, a function pointer to a function, which points to a function which takes some variable /value of type int as an argument and returns a

	pointer to a function which accepts some value /variable of type double and returns a pointer
int (*(*(*f3)(int))(double))(float)	--f3, a function pointer to a function, which points to a function which takes some variable /value of type int as an argument and returns a pointer to a function which accepts some value /variable of type double and returns a function pointer to a function which accepts some value/ variable of float datatype as an argument and returns a value of type int

12) Answer is a) 1 2.000000

Explanation: Consider the declaration at line 2: void (*(f)())(int, float);

f is a function which returns a pointer to a function which takes two variables/values as argument: one of int datatype and another of float datatype and returns void or nothing

Now, consider line 3: void (*(*x)())(int, float) = f;

void (*(*x)())(int, float) x is a pointer to a function which returns a function pointer to a function which takes two variables/values as argument: one of int datatype and another of float datatype and returns void or nothing

and the function f is assigned to function pointer x

Now, consider line 4: void ((*y)(int, float));

y is a function pointer to a function which takes two values as arguments: one of int datatype, another is of float datatype and returns nothing/void

Now, at line 8, y is initialized to the function pointer returned by function f which is invoked (function f) through function pointer x

Now, since, the function f returns a function pointer to a function which takes two values as arguments: one of int datatype, another is of float datatype and returns nothing/void, it can be successfully assigned to the function pointer y

Now, foo is such a function which takes two values as arguments: one of int datatype, another is of float datatype and returns nothing/void. So, foo can be returned by function f and since the function pointer returned by function f is assigned to function pointer y, y can be used to invoke foo

So, no compilation error will be caused.

Program will print 1 2.000000

13) Answer is a) Compile time error.

Explanation: Compiler would say the following things:

- expected identifier or ‘(’ before ‘=’ token x = f at line 7
- expected identifier or '(' before '=' token x()

14) Answer is d) Nothing.

Explanation:

Consider void (*(f)())(int, float): this declaration at line 2

we declare a function f which returns a pointer to a function taking arguments of type int and float and returning nothing/void

Consider typedef void (*(*x)())(int, float): this declaration at line 3

We basically create a type 'x' for funtion pointers which point to function taking arguments of int and float datatype and return void/nothing

Now, at line line 7, we declare a variable p of type x, and the function f is assigned to it

Now, as the function f and the function pointer p of type x are compatiable, this will cause no error.

Now, f is called using p (of type x) which returns a function pointer a function taking arguments of type int and float and returning nothing/void (foo is such a function)

But, the function pointer returned by f (which is called/invoked by p of type x) is not assigned to any function pointer to call foo.

So, this will print nothing.

17) Answer is b) ptr is pointer to function passing int returning void

18) Answer is b) int *(*ptr[3])(): Here ptr is an array of size 3 or of 3 elements which are pointers to function taking nothing and returning int.

19) Answer is b) int **ptr: Here, ptr is a pointer to an int pointer

20) Answer is a) char *str[5] : Here str is an array of pointers of type character

21) Answer is d) Both i) and ii) will work legal and flawlessly

Additional Note: However, it is also true, they cannot exist in the same scope due to same name. (the function pointer ptr and the array of pointer elements which point to character sequences :ptr)

22) Answer is a) Compile time error. Since new structure type student declaration does not end with a semicolon.

23) Answer is b) Compile time error due to presence of initialization statement int no=5 within the new structure type student declaration

24) Answer is c) Compile time error.

Line 9 should be corrected to struct student s to make the program compiled and run successfully

25) Answer is d) 8
26) Answer is b) False

References:

1) http://www.sanfoundry.com/c-interview-questions-answers/
2) http://stackoverflow.com/