REF.

https://www.geeksforgeeks.org/declare-a-cc-function-returning-pointer-to-array-of-integers/

* Difficulty Level : [Hard](https://www.geeksforgeeks.org/hard/)
* Last Updated : 29 May, 2017

Declare “a function with argument of int\* which returns pointer to an array of 4 integer pointers”.

At the first glance it may look complex, we can declare the required function with a series of decomposed statements.

1. We need, a function with argument int \*,

function(int \*)

2. a function with argument int \*, returning pointer to

(\*function(int \*))

3. a function with argument int \*, returning pointer to array of 4

(\*function(int \*))[4]

4. a function with argument int \*, returning pointer to array of 4 integer pointers

int \*(\*function(int \*))[4];

How can we ensure that the above declaration is correct? The following program can cross checks our declaration,

|  |
| --- |
| #include<stdio.h>    // Symbolic size  #define SIZE\_OF\_ARRAY (4)    // pointer to array of (SIZE\_OF\_ARRAY) integers  typedef int \*(\*p\_array\_t)[SIZE\_OF\_ARRAY];    // Declaration : compiler should throw error  // if not matched with definition  int \*(\*function(int \*arg))[4];    // Definition  : 'function' returning pointer to an  // array of integer pointers  p\_array\_t function(int \*arg)  {     // array of integer pointers     static int \*arr[SIZE\_OF\_ARRAY] = {NULL};       // return this     p\_array\_t pRet = &arr;       return pRet;  }    int main()  {  } |

The macro SIZE\_OF\_ARRAY is used for symbolic representation of array size. p\_array\_t is typedefined as “pointer to an array of 4 integers”. If our declaration is wrong, the program throws an error at the ‘function‘ definition.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.