**1. What is the meaning of the following declaration?**

**int(\*p[5])();**

a) p is pointer to function  
b) p is array of pointer to function  
c) p is pointer to such function which return type is the array  
d) p is pointer to array of function

Answer: b  
Explanation: In the above declaration the variable p is the array, not the pointer.

2. What is size of generic pointer in C++ (in 32-bit platform)?  
a) 2  
b) 4  
c) 8  
d) 0

Answer: b  
Explanation: Size of any type of pointer is 4 bytes in 32-bit platforms.

3. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int a[2][4] = {3, 6, 9, 12, 15, 18, 21, 24};
6. cout << \*(a[1] + 2) << \*(\*(a + 1) + 2) << 2[1[a]];
7. return 0;
8. }

a) 15 18 21  
b) 21 21 21  
c) 24 24 24  
d) Compile time error

Answer: b  
Explanation: a[1][2] means 1 \* (4)+2 = 6th element of an array starting from zero.  
Output:

$ g++ point.cpp

$ a.out

21 21 21

4. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int i;
6. const char \*arr[] = {"C", "C++", "Java", "VBA"};
7. const char \*(\*ptr)[4] = &arr;
8. cout << ++(\*ptr)[2];
9. return 0;
10. }

a) ava  
b) java  
c) c++  
d) compile time error

Answer: a  
Explanation: In this program we are moving the pointer from first position to second position and printing the remaining value.  
Output:

$ g++ point1.cpp

$ a.out

ava

5. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int arr[] = {4, 5, 6, 7};
6. int \*p = (arr + 1);
7. cout << \*p;
8. return 0;
9. }

a) 4  
b) 5  
c) 6  
d) 7

Answer: b  
Explanation: In this program, we are making the pointer point to next value and printing it.

$ g++ point3.cpp

$ a.out

5

6. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int arr[] = {4, 5, 6, 7};
6. int \*p = (arr + 1);
7. cout << arr;
8. return 0;
9. }

a) 4  
b) 5  
c) address of arr  
d) 7

Answer: c  
Explanation: As we counted to print only arr, it will print the address of the array.  
Output:

$ g++ point2.cpp

$ a.out

0xbfb1cff

7. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main ()
4. {
5. int numbers[5];
6. int \* p;
7. p = numbers; \*p = 10;
8. p++; \*p = 20;
9. p = &numbers[2]; \*p = 30;
10. p = numbers + 3; \*p = 40;
11. p = numbers; \*(p + 4) = 50;
12. for (int n = 0; n < 5; n++)
13. cout << numbers[n] << ",";
14. return 0;
15. }

a) 10,20,30,40,50,  
b) 1020304050  
c) compile error  
d) runtime error

Answer: a  
Explanation: In this program, we are just assigning a value to the array and printing it and immediately dereferencing it.  
Output:

$ g++ point4.cpp

$ a.out

10,20,30,40,50,

8. What will be the output of the following C++ code?

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int arr[] = {4, 5, 6, 7};
6. int \*p = (arr + 1);
7. cout << \*arr + 9;
8. return 0;
9. }

a) 12  
b) 5  
c) 13  
d) error

Answer: c  
Explanation: In this program, we are adding the value 9 to the initial value of the array, So it’s printing as 13.  
Output:

$ g++ point5.cpp

$ a.out

13

**Taking Input for Two Dimensional Array**

#include <iostream>

using namespace std;

int main() {

int numbers[2][3];

cout << "Enter 6 numbers: " << endl;

// Storing user input in the array

for (int i = 0; i < 2; ++i) {

for (int j = 0; j < 3; ++j) {

cin >> numbers[i][j];

}

}

cout << "The numbers are: " << endl;

// Printing array elements

for (int i = 0; i < 2; ++i) {

for (int j = 0; j < 3; ++j) {

cout << "numbers[" << i << "][" << j << "]: " << numbers[i][j] << endl;

}

}

return 0;

}

**Output**

Enter 6 numbers:

1

2

3

4

5

6

The numbers are:

numbers[0][0]: 1

numbers[0][1]: 2

numbers[0][2]: 3

numbers[1][0]: 4

numbers[1][1]: 5

numbers[1][2]: 6

### Q10. Changing Value Pointed by Pointers

#include <iostream>

using namespace std;

int main() {

int var = 5;

int\* pointVar;

// store address of var

pointVar = &var;

// print var

cout << "var = " << var << endl;

// print \*pointVar

cout << "\*pointVar = " << \*pointVar << endl

<< endl;

cout << "Changing value of var to 7:" << endl;

// change value of var to 7

var = 7;

// print var

cout << "var = " << var << endl;

// print \*pointVar

cout << "\*pointVar = " << \*pointVar << endl

<< endl;

cout << "Changing value of \*pointVar to 16:" << endl;

// change value of var to 16

\*pointVar = 16;

// print var

cout << "var = " << var << endl;

// print \*pointVar

cout << "\*pointVar = " << \*pointVar << endl;

return 0;

}

**Output**

var = 5

\*pointVar = 5

Changing value of var to 7:

var = 7

\*pointVar = 7

Changing value of \*pointVar to 16:

var = 16

\*pointVar = 16