**1.Explain the result of following program**

#include <iostream>

using namespace std;

typedef int \*IntPtrType;

void main()

{

IntPtrType ptr\_a, ptr\_b, \*ptr\_c;

ptr\_a = new int;// **Dynamic memory allocation**  for pointer ptr\_a

\*ptr\_a = 3; // Assign address vairble 3 for pointer ptr\_a

ptr\_b = ptr\_a;// Assign address pointer ptr\_a is pointting for pointer ptr\_b

cout << \*ptr\_a << " " << \*ptr\_b << "\n";// print out value of pointer ptr\_a = 3 and ptr\_b =3

ptr\_b = new int;// **Dynamic memory allocation**  for pointer ptr\_b

\*ptr\_b = 9; // Assign again value 9 for pointer ptr\_b

cout << \*ptr\_a << " " << \*ptr\_b << "\n";// print out value of pointer ptr\_a = 3 and ptr\_b =9

\*ptr\_b = \*ptr\_a; // Assign value of pointer ptr\_a = 3 for pointer ptr\_b;

cout << \*ptr\_a << " " << \*ptr\_b << "\n";// print out value of pointer ptr\_a = 3 and ptr\_b =3

delete ptr\_a;//Delete link between pointer ptr\_a and value pointer ptr\_a is pointting.

ptr\_a = ptr\_b;// Assign address pointer ptr\_b for pointer ptr\_a

cout << \*ptr\_a << " " << \*ptr\_b << "\n";// print out value of pointer ptr\_a = 3 and ptr\_b =3

ptr\_c = &ptr\_a;//Assign address of pointer ptr\_a for pointer ptr\_c

cout << \*ptr\_c << " " << \*\*ptr\_c << "\n";// printf out address of pointer \*ptr\_a = 0x1519f0 and its value is =3( Because pointer ptr\_c is pointer to pointer

delete ptr\_a;// Delete link between pointer ptr\_a and value pointer ptr\_a is pointting.

ptr\_a = NULL;//Assign value for pointer ptr\_a = NULL system("pause");

}

**2.Detect and solve problems of following program**

The problem here is we not yet **Dynamic memory allocation**  for vairble pointer p and we miss operator [] after delete a.

#include <iostream>

using namespace std;

void main()

{

int a[4] = { 1, 2, 3, 4 };

int \*p = new int; // Solve is a **Dynamic memory allocation**  for pointer p.

p = a;//assign address vairble a for pointer p

int \*p2 = new int;

delete p;

delete[] a; // add operator [] for delete array.

delete p2;

}