Exercises of pointers and reference

```cpp
#include <iostream>
using namespace std;

int main()
{
    int num= 20, num1= 77;
    int &renum= num;
    renum=num1;
    cout<<"renum: "<<renum<<" num: "<<num<< " num1: "<<num<<endl;
    return 0;
}

```

Output: renum: 77 num: 77 num1: 77

*Note: “renum” and “num” has the same address and never change. And this also applies on “renum1” and “num1”.

```cpp
#include <iostream>
using namespace std;

int main()
{
    int num= 20, num1= 77;
    int &renum= num;
    int &renum1 = num1;
    renum=renum1;
    renum1=90;
    cout<<"renum: "<<renum<<" num: "<<num<<
    " num1: "<<num<<" renum1: "<<renum1<<endl;
    return 0;
}
```

Output: renum: 77 num: 77 num1: 77 renum1: 90

*Important: You need to initialize the reference during declaration. And you can't change its reference. Remember: pointer "refnum" and its reference "num" must have the same variable type.

```cpp
#include <iostream>
using namespace std;

int main()
{
    int num= 20;
    int *p= &num;
    cout<<"num: "<<num<<" p: "<<p
    <<" *p: "<<*p<<endl;
    return 0;
}
```

Output: num: 20 p: 0x7ffeefbff5c8 *p: 20

*Note: when we want to print the value of pointer “p” there is two way : print “num”, or print *p, without * “asterisk” it will be print the address of num.

```cpp
#include <iostream>
using namespace std;

int main()
{
    int x= 10;
    int *p= &x;
    cout<<"adress x: "<<&x
    <<" adress p: "<<&p<<endl;
    int y=15;
    int &ref= y;
    cout<<"adress y: "<<&y<<
    " adress ref: "<<&ref<<endl;
    return 0;
}
```

Output:

adress x: 0x7ffeefbff5c8 adress p: 0x7ffeefbff5c0

adress y: 0x7ffeefbff5bc adress ref: 0x7ffeefbff5bc

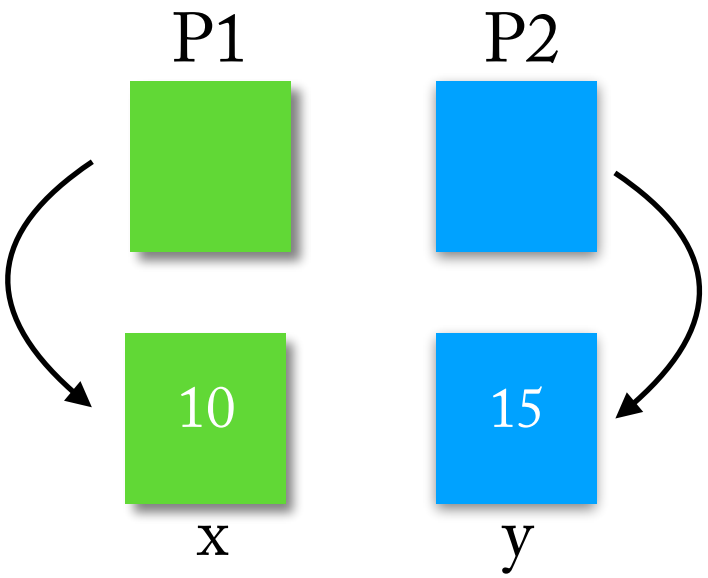
*Note: the difference between pointers and reference is the pointer “p” has the different address to its value reference “x”, but reference “ref” has the same address of its value “y”.

```cpp
#include <iostream>
using namespace std;

int main()
{
    int x= 10;
    cout<<*(&x)<<endl;


}
```

Output: 10

*it will be go to the address of x and find its value <:

```cpp
#include <iostream>
using namespace std;

int main()
{
    int *p1, *p2;
    int x=10, y=15;
    p2= &y;
    p1= &x;
    *p1= *p2;
    cout<<"p1= "<<*p1<<" p2= "<<*p2<<endl;
    cout<<"x: "<<x<<" y: "<<y<<endl;
}
```

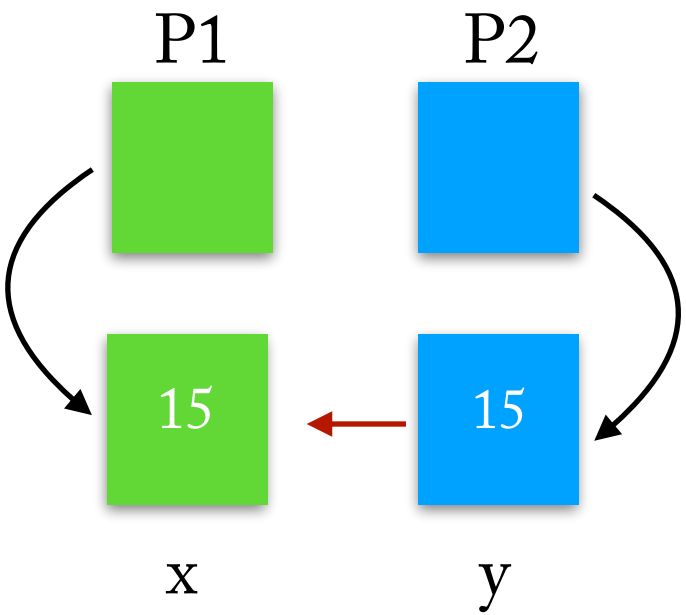


Output:

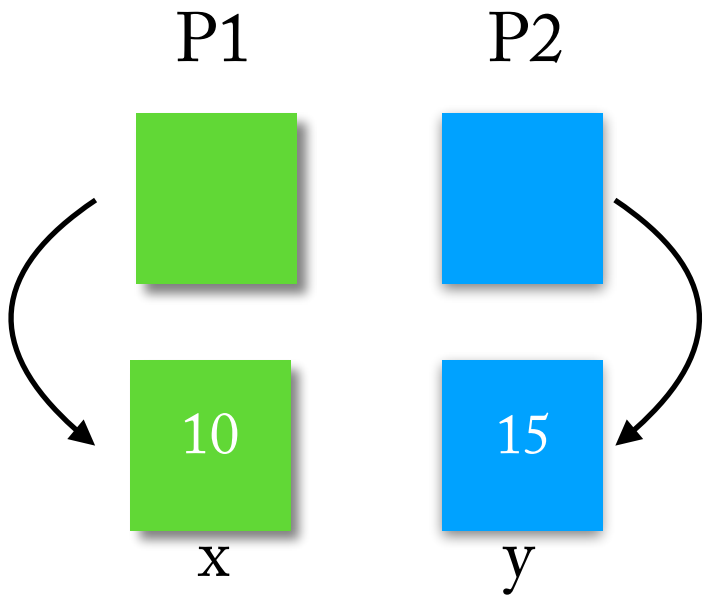
p1= 15 p2= 15

x: 15 y: 15

Transfer value “p2” to “p1”



```cpp
#include <iostream>
using namespace std;

int main()
{
    int *p1, *p2;
    int x=10, y=15;
    p2= &y;
    p1= &x;
    p1= p2;
    cout<<"p1= "<<*p1<<" p2= "<<*p2<<endl;
    cout<<"x: "<<x<<" y: "<<y<<endl;
}
```

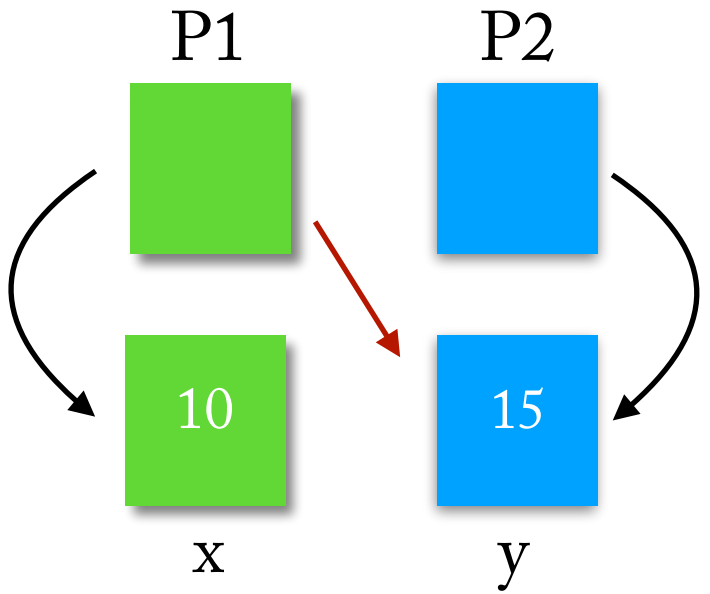


Output:

p1= 15 p2= 15

x: 10 y: 15

Make “p1” pointer to y



```
2  #include <iostream>
3  using namespace std;
4  int main( ) {
5      int *p1;
6      p1= new int;
7      *p1= 20;
8      cout<<"p1= "<<*p1<<endl;
9      p1= NULL;
10     cout<<"p1= "<<*p1<<endl;
11
12 }
```

Output:

p1 = 20

p1 = (lldb)

We can't use pointer "p1" after delete it.

```cpp
#include <iostream>
using namespace std;
double *GetSalary() {
    double salary= 10.5;
    double *HourlySalary = &salary;
    return HourlySalary ;

}
int main( ) {
    double salary= *GetSalary();
    cout << "Hourly Salary: " << salary << endl;
    return 0;
}
```

Output:
Hourly salary: 10.5

It's safe to use it here because we return the address and save it in a pointer that local in the main

```cpp
#include <iostream>
using namespace std;
double * GetSalary(){
    double salary =10.50;
    double *HourlySalary = &salary;
    return HourlySalary ;
}
int
main(){
    double * salary = GetSalary();
    cout<< "Hourly Salary: " << *salary << endl;
    return 0;
}
```

Output: garbage value.

It is not safe to use it because we return the address of salary and save it in a pointer that is local in the main, so once we leave the function “Getsalary” the address of salary we can’t use it anymore.

```cpp
#include <iostream>
using namespace std;
int main( ) {
    int i=10, j=20;
    int *p1= &i;
    int *p2= &j;
    *p1= *p1 + 5;
    *p2 = *p2 + *p1;
    cout<<"*p1= "<<*p1<<"  *p2= "<<*p2
    <<endl;
}
```

Output:

***p1 = 15 *p2= 35**

```
#include <iostream>
using namespace std;
int main( ) {
    int i=10, j=20;
    int *p1= &i;
    int *p2= &j;
    *p1 = ++*p1;
    cout<<"*p1= "<<*p1
    <<endl;
    *p1= *p2++;
    cout<<"*p1= "<<*p1
    <<endl;

}
```

Output:

***p1 = 11**

***p2= 20**