# Functions and Pointers – Q1

Assume the following declaration:

```
int number;
int *p;
```

Assume also that the address of number is 7700 and the address of p is 3478.

| | | |
|---|---|---|
| 3478 | | p |
| | . . . | |
| 7700 | | number |

For each case below, determine the value of

**(a) number (b) &number (c) p (d) &p (e) *p**

All of the results are cumulative.

**(i) p = 100; number = 8**

**(ii) number = p**

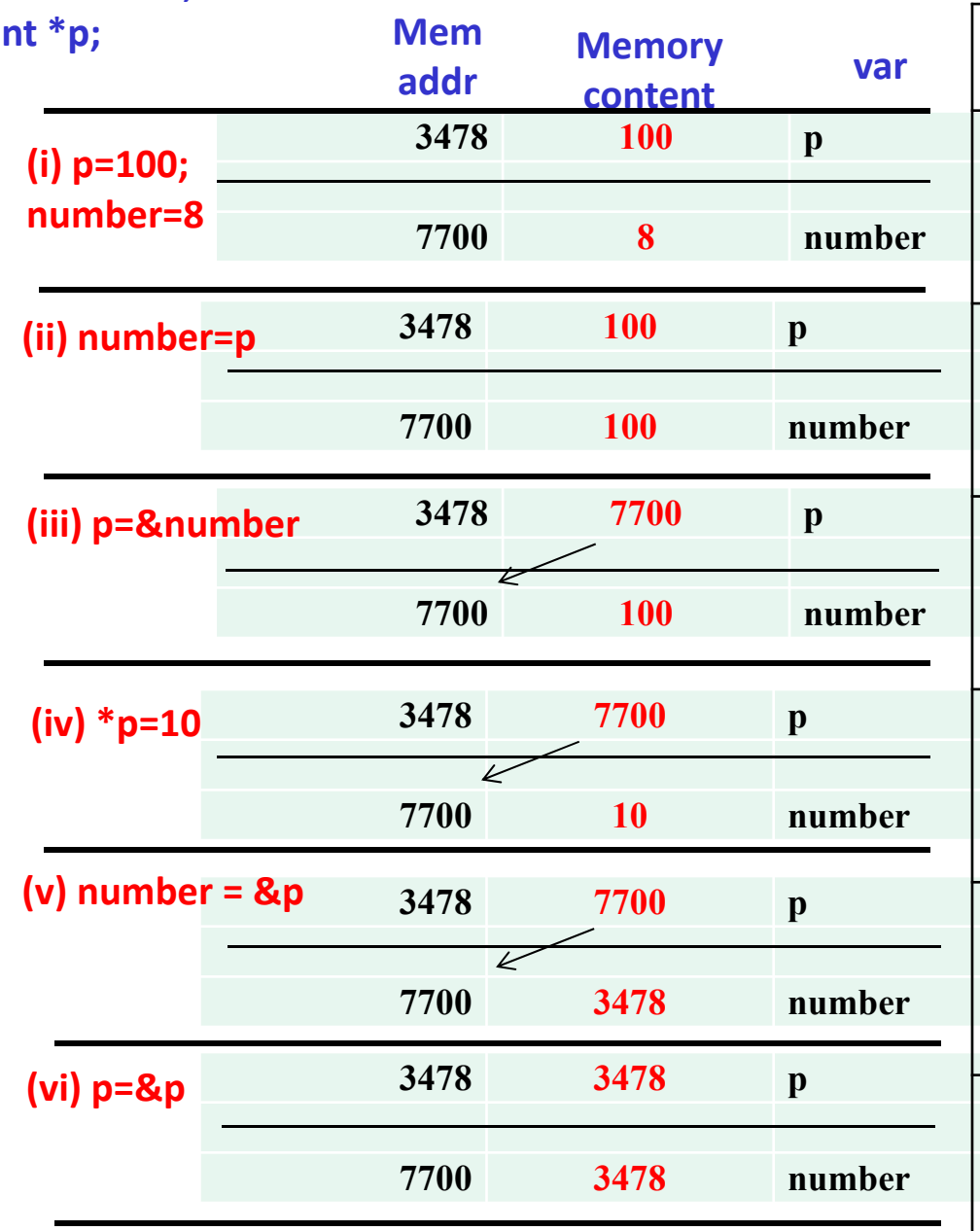
**(iii) p = &number**

**(iv) *p = 10**

**(v) number = &p**

**(vi) p = &p**

int number;
int *p;




| (a) num | (b) &num | (c) p | (d) &p | (e) *p |
|---|---|---|---|---|
| 8 | 7700 | 100 | 3478 | Content of mem location 100 |
| 100 | 7700 | 100 | 3478 | Content of mem location 100 |
| 100 | 7700 | 7700 | 3478 | 100 |
| 10 | 7700 | 7700 | 3478 | 10 |
| 3478 | 7700 | 7700 | 3478 | 3478 |
| 3478 | 7700 | 3478 | 3478 | 3478 |

# Functions and Pointers – Q2

Write a function that counts the number of digits for a non-negative integer. For example, 1234 has 4 digits. The function **numDigits1()** returns the result. The function prototype is given below:

```
int numDigits1(int num);      // call by value
```

Write another function **numDigits2()** that passes the result through the second parameter, *result*. The function prototype is given below:

```
void numDigits2(int num, int *result);  // call by reference
```

Write a C program to test the functions.

Some sample input and output sessions are given below:

(1) Enter the number:
*1234*
numDigits1(): 4

(2) Enter the number:
*13579*
numDigits2(): 5

**Note: When programming with number, use % operator to get the remainder of a number, and / operator to get the quotient of the number.**

**For example: 1234/10 -> 123; 1234%10 -> 4**

# Q2 – Call by Value

```
#include <stdio.h>
int numDigits1(int num);
int main()
{
   int number;

   printf("Enter the number: \n");
   scanf("%d", &number);
   printf("numDigits1(): %d\n",
           numDigits1(number));
   return 0;
}
int numDigits1(int num)
{
   int count = 0;
   do {
      count++;
      num = num/10;
   } while (num > 0);
   return count;
}
```


number


num

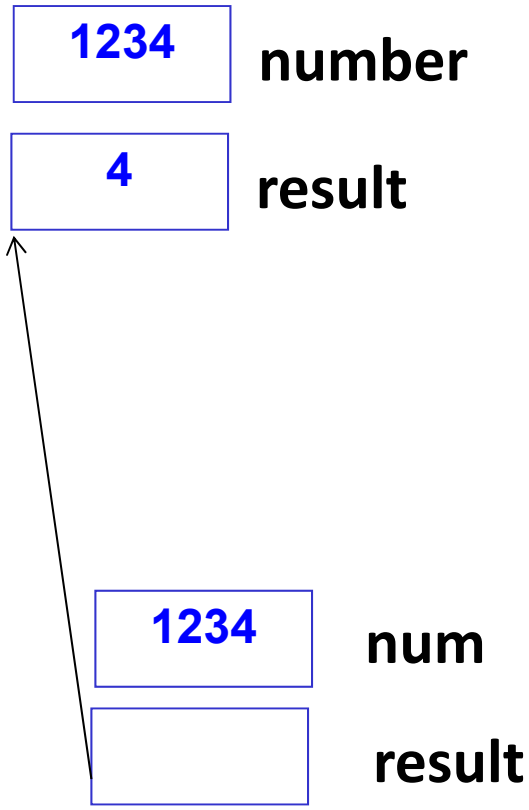

count

# Q2 – Call by Reference

```
#include <stdio.h>
void numDigits2(int num, int *result);
int main()
{
   int number, result=0;

   printf("Enter the number: \n");
   scanf("%d", &number);
   numDigits2(number, &result);
   printf("numDigits2(): %d\n", result);

   return 0;
}
void numDigits2(int num, int *result)
{
   *result=0;
   do {
      (*result)++;
      num = num/10;
   } while (num > 0);
}
```



# Functions and Pointers – Q3

Write the function **digitPos1()** that returns the position of the first appearance of a specified digit in a positive number. The position of the digit is counted from the right and starts from 1. If the required digit is not in the number, the function should return 0. For example, digitPos1(12315, 1) returns 2 and digitPos1(12, 3) returns 0. The function prototype is given below:

```
int digitPos1(int num, int digit);    // call by value
```

Write another function **digitPos2()** that passes the result through the third parameter, *result*. For example, if num = 12315 and digit = 1, then *result = 2 and if num=12 and digit = 3, then *result = 0. The function prototype is given below:

```
void digitPos2(int num, int digit, int *result);    // call by reference
```

Write a C program to test the functions.

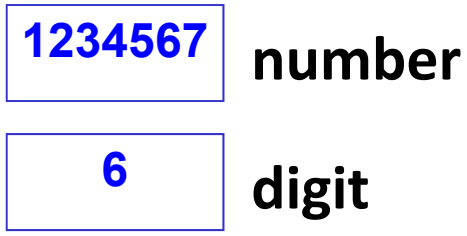
Some sample input and output sessions are given below:

```
(1)     Enter the number:
        1234567
        Enter the digit:
        6
        digitPos1(): 2
(2)     Enter the number:
        1234567
        Enter the digit:
        8
        digitPos2(): 0
```

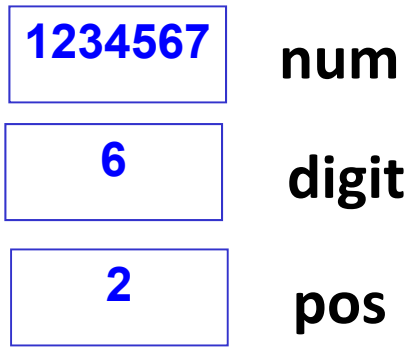
# Q3 – Call by Value

```
#include <stdio.h>
int digitPos1(int num, int digit);
int main()
{
   int number, digit;

   printf("Enter the number: \n");
   scanf("%d", &number);
   printf("Enter the digit: \n");
   scanf("%d", &digit);
   printf("digitPos1(): %d\n",
      digitPos1(number, digit));
   return 0;
}
```



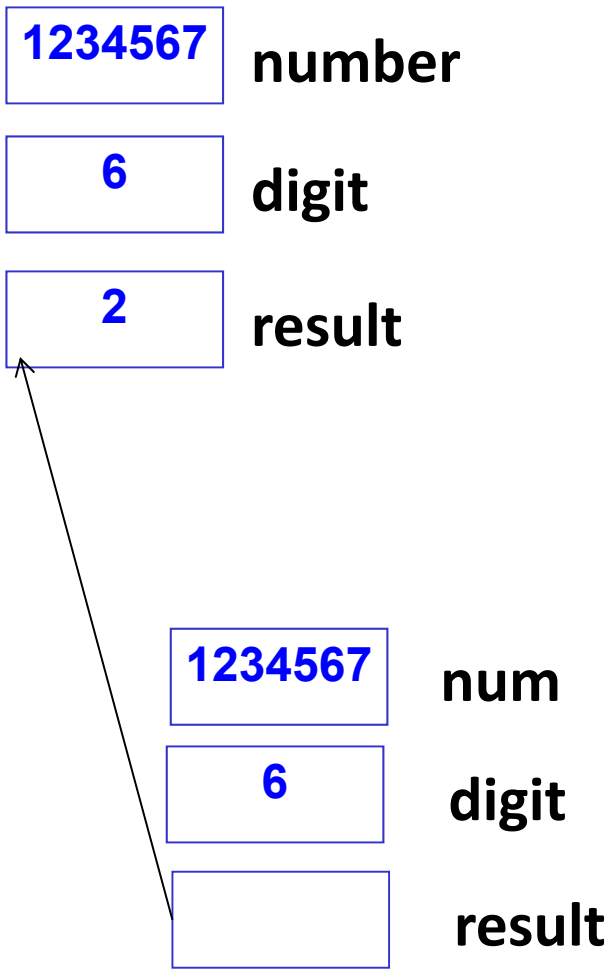

```
int digitPos1(int num, int digit)
{
   int pos=0;
   do {
      pos++;
      if (num % 10 == digit)
         return pos;
      num = num / 10;
   } while (num > 0);
   return 0;
}
```



# Q3 – Call be Reference

```
#include <stdio.h>
void digitPos2(int num,int digit,int *result);
int main()
{
   int number, digit, result=0;
   printf("Enter the number: \n");
   scanf("%d", &number);
   printf("Enter the digit: \n");
   scanf("%d", &digit);
   digitPos2(number, digit, &result);
   printf("digitPos2(): %d\n", result);
   return 0;
}
```

```
void digitPos2(int num, int digit,
int *result)
{
   int pos=0;
   *result=0;
   do {
     pos++;
      if (num % 10 == digit){
         *result = pos;
         break;
      }
      num = num / 10;
   } while (num > 0);
}
```



# Functions and Pointers – Q4

Write a function **square1()** that returns the square of a positive integer number *num*, by computing the sum of odd integers starting with 1 as shown in the example below. The result is returned to the calling function. For example, if *num* = 4, then $4^2 = 1 + 3 + 5 + 7 = 16$ is returned; if *num* = 5, then $5^2 = 1 + 3 + 5 + 7 + 9 = 25$ is returned. The function prototype is:

```
int square1(int num);          // call by value
```

Write another function **square2()** that passes the result through the third parameter, *result*. For example, if *num* = 4, then *result = $4^2 = 1 + 3 + 5 + 7 = 16$; if *num* = 5, then *result = $5^2 = 1 + 3 + 5 + 7 + 9 = 25$. The function prototype is:

```
void square2(int num, int *result);     // call by reference
```

A sample input and output session is given below:

```
(1)     Enter a number:
        4
        square1(): 16
(2)     Enter a number:
        5
        square2(): 25
```

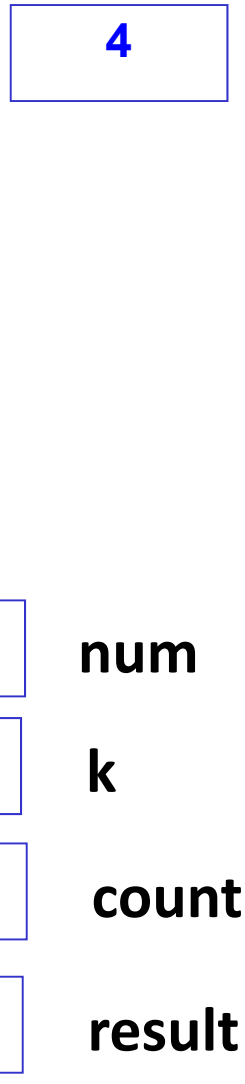
# Q4 – Call by Value

```
#include <stdio.h>
int square1(int num);
int main()
{
   int number;

   printf("Enter the number: \n");
   scanf("%d", &number);
   printf("square1(): %d\n",
          square1(number));
   return 0;
}
```

```
int square1(int num)
{
   int count=0, k=1, result=0;

   while (count < num)
   {
      result += k;
      k += 2;
      count++;
   }
   return result;
}
```



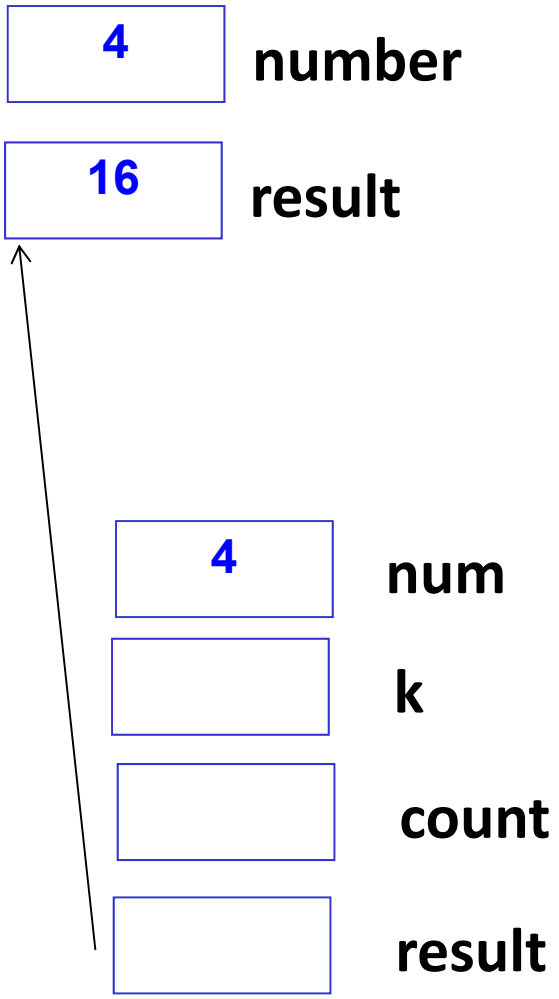
# Q4 – Call by Reference

```
#include <stdio.h>
void square2(int num, int *result);
int main()
{
   int number, result=0;

   printf("Enter the number: \n");
   scanf("%d", &number);
   square2(number, &result);
   printf("square2(): %d\n", result);
   return 0;
}
```

```
void square2(int num, int *result)
{
   int count=0, k=1;

   *result=0;
   while (count < num)
   {
      *result += k;
      k += 2;
      count++;
   }
}
```



# Functions and Pointers – Q5

What will be the output of the following program?

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2( int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

**(1) h = 5, k = 15  line (i)**

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2(int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

(1) h = 5, k = 15  line (i)

**(2) h = -100, k = -100 line (v)**

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2(int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

(1) h = 5, k = 15  line (i)

**(3) h = 5, k = 15  line (ii)**

(2) h = -100, k = -100 line (v)

```c
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2(int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

(1) h = 5, k = 15  line (i)

(3) h = 5, k = 15  line (ii)

(2) h = -100, k = -100 line (v)

**(4) h = 5, k = 15  line (vi)**

**(5) h = 100, k = 100 line (vii)**

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2(int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

(1) h = 5, k = 15  line (i)

(3) h = 5, k = 15  line (ii)

**(6) h = 5, k = 15  line (iii)**

(2) h = -100, k = -100 line (v)

(4) h = 5, k = 15  line (vi)

(5) h = 100, k = 100 line (vii)

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2(int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

(1) h = 5, k = 15  line (i)

(3) h = 5, k = 15  line (ii)

(6) h = 5, k = 15  line (iii)

(2) h = -100, k = -100 line (v)

(4) h = 5, k = 15  line (vi)

(5) h = 100, k = 100 line (vii)

**(7) h = 5, k = 15  line (viii)**

**(8) h = 200, k = 200 line (ix)**

```
#include  <stdio.h>
void function0();
void function1(int h, int k);
void function2(int *h, int *k);
int main(){
   int h, k;
   h = 5;
   k = 15;
   printf("h = %d, k = %d\n", h, k);   /* line (i) */
   function0();
   printf("h = %d, k = %d\n", h, k);   /* line (ii) */
   function1(h, k);
   printf("h = %d, k = %d\n", h, k);   /* line (iii) */
   function2(&h, &k);
   printf("h = %d, k = %d\n", h, k);   /* line (iv) */
   return 0;
}
void function0(){
   int h, k;
   h = k = -100;
   printf("h = %d, k = %d\n", h, k);   /* line (v) */
}
void function1( int h, int k){
   printf("h = %d, k = %d\n", h, k);   /* line (vi) */
   h = k = 100;
   printf("h = %d, k = %d\n", h, k);   /* line (vii) */
}
void function2(int *h, int *k){
   printf("h = %d, k = %d\n", *h, *k); /* line (viii) */
   *h = *k = 200;
   printf("h = %d, k = %d\n", *h, *k); /* line (ix) */
}
```

(1) h = 5, k = 15  line (i)

(3) h = 5, k = 15  line (ii)

(6) h = 5, k = 15  line (iii)

**(9) h = 200, k = 200 line (iv)**

(2) h = -100, k = -100 line (v)

(4) h = 5, k = 15  line (vi)

(5) h = 100, k = 100 line (vii)

**(7) h = 5, k = 15  line (viii)**

**(8) h = 200, k = 200 line (ix)**