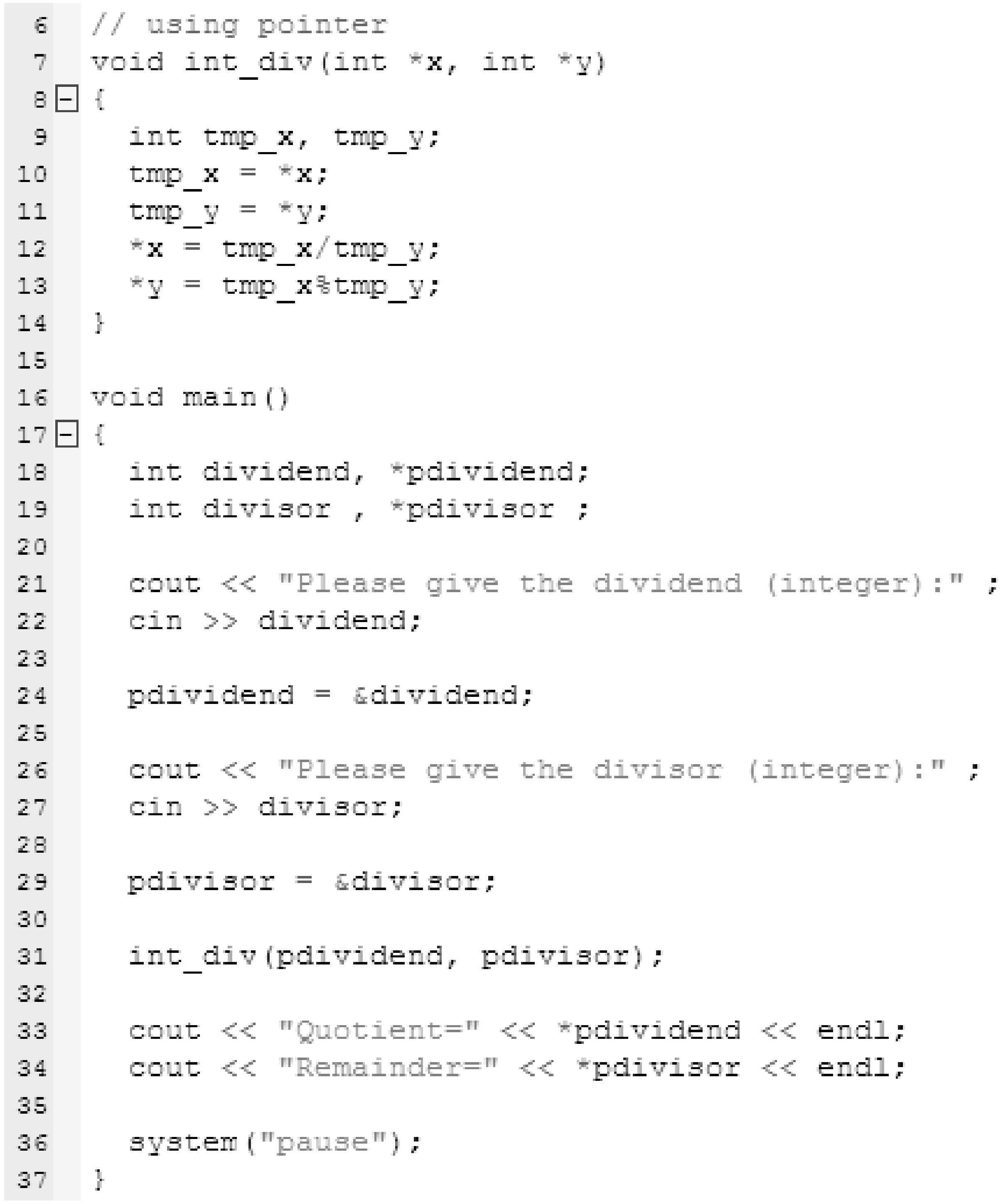
10

Case Study of Pointers (part 1)

Study Case: Return Q & R of a division simultaneously

- Problem definition
  - Use *pointer* to return two values in a defined *function* simultaneously
  - Use integer division as an example
    - Implement a function for integer division
    - Return the Q & R values simultaneously
    - Q: Quotient
    - R: Remainder



```
// using pointer
void int_div(int *x, int *y)
{
  int tmp_x, tmp_y;
  tmp_x = *x;
  tmp_y = *y;
  *x = tmp_x/tmp_y;
  *y = tmp_x%tmp_y;
}

void main()
{
  int dividend, *pdividend;
  int divisor , *pdivisor ;

  cout << "Please give the dividend (integer):" ;
  cin >> dividend;

  pdividend = &dividend;

  cout << "Please give the divisor (integer):" ;
  cin >> divisor;

  pdivisor = &divisor;

  int_div(pdividend, pdivisor);

  cout << "Quotient=" << *pdividend << endl;
  cout << "Remainder=" << *pdivisor << endl;

  system("pause");
}
```

Case 2: `Pointer` v.s. `Reference`

- **Please complete the following codes to compare call-by-value, call-by-pointer, and call-by-reference**
  - **Use the similar ways of the following two examples**

```
void cubeByPtr(int *nPtr);

int main(void) {
  int number = 5;
  ......
  cubeByPtr(&number);
  ......
}

void cubeByPtr(int *nPtr) {
  (*nPtr)=*nPtr**nPtr**nPtr;
}
```

```
void cubeByRef(int &nRef);

int main(void) {
  int number = 5;
  ......
  cubeByRef(number);
  ......
}

void cubeByRef(int &nRef) {
  nRef = nRef*nRef*nRef;
}
```

Outline

fig07_06.c

```c
/* Fig. 7.6: fig07_06.c
   Cube a variable using call-by-value */
#include <stdio.h>

int cubeByValue( int n ); /* prototype */

int main( void )
{
   int number = 5; /* initialize number */

   printf( "The original value of number is %d", number );

   /* pass number by value to cubeByValue */
   number = cubeByValue( number );

   printf( "\nThe new value of number is %d\n", number );

   return 0; /* indicates successful termination */

} /* end main */

/* calculate and return cube of integer argument */
int cubeByValue( int n )
{
   return n * n * n;   /* cube local variable n and return result */

} /* end function cubeByValue */
```

```
The original value of number is 5
The new value of number is 125
```

Outline

fig07_07.c

```c
/* Fig. 7.7: fig07_07.c
   Cube a variable using call-by-reference with a pointer argument */

#include <stdio.h>

void cubeByReference( int *nPtr ); /* prototype */

int main( void )
{
   int number = 5; /* initialize number */

   printf( "The original value of number is %d", number );

   /* pass address of number to cubeByReference */
   cubeByReference( &number );

   printf( "\nThe new value of number is %d\n", number );

   return 0; /* indicates successful termination */

} /* end main */

/* calculate cube of *nPtr; modifies variable number in main */
void cubeByReference( int *nPtr )
{
   *nPtr = *nPtr * *nPtr * *nPtr;  /* cube *nPtr */
} /* end function cubeByReference */
```

```
The original value of number is 5
The new value of number is 125
```