extEvenDigits

Write a function that extracts the even digits from a positive number, and combines the even digits sequentially into a new number. The new number is returned to the calling function. If the input number does not contain any even digits, then the function returns -1. For example, if the input number is 1234567, then 246 will be returned; and if the input number is 13, then –1 will returned. You do not need to consider the input number such as 0, 10, 3033, etc. Write the function in two versions. The function extEvenDigits1() returns the result to the caller, while the function extEvenDigits2() returns the result through the pointer parameter, result. The function prototypes are given as follows:

```
int extEvenDigits1(int num);
void extEvenDigits2(int num, int *result);
```

A sample program template is given below to test the functions:

```
#include <stdio.h>
#define INIT_VALUE 999
int extEvenDigits1(int num);
void extEvenDigits2(int num, int *result);
int main()
{
  int number, result = INIT_VALUE;

  printf("Enter a number: \n");
  scanf("%d", &number);
  printf("extEvenDigits1(): %d\n", extEvenDigits1(number));
  extEvenDigits2(number, &result);
  printf("extEvenDigits2(): %d\n", result);
  return 0;
}
int extEvenDigits1(int num)
{
  /* Write your code here */
}
void extEvenDigits2(int num, int *result)
{
  /* Write your code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:
Enter a number:
1234
extEvenDigits1(): 24
extEvenDigits2(): 24

(2) Test Case 2:
Enter a number:
1357

extEvenDigits1(): -1
extEvenDigits2(): -1

(3) Test Case 3:
Enter a number:
2468
extEvenDigits1(): 2468
extEvenDigits2(): 2468

(4) Test Case 4:
Enter a number:
6
extEvenDigits1(): 6
extEvenDigits2(): 6