ASSIGNMENT - RECURSIVE FUNCTIONS

1. (**rCountEvenDigits**) Write a **recursive** C function that counts the number of even digits in a specified positive number (bigger than 0), *num*. For example, if `num` is 105006, then the function will return 4; and if `num` is 1357, the function will return 0. Write the recursive function in two versions. The function **rCountEvenDigits1()** computes and returns the result. The function **rCountEvenDigits2()** passes the result through the pointer parameter, `result`. The function prototypes are given as follows:

```
int rCountEvenDigits1(int num);
void rCountEvenDigits2(int num, int *result);
```

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

```
#include <stdio.h>
int rCountEvenDigits1(int num);
void rCountEvenDigits2(int num, int *result);
int main()
{
   int number, result;

   printf("Enter the number: \n");
   scanf("%d", &number);
   printf("rCountEvenDigits1(): %d\n", rCountEvenDigits1(number));
   rCountEvenDigits2(number, &result);
   printf("rCountEvenDigits2(): %d\n", result);
   return 0;
}
int rCountEvenDigits1(int num)
{
   /* Write your program code here */
}
void rCountEvenDigits2(int num, int *result)
{
   /* Write your program code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:
```
Enter the number:
105006
rCountEvenDigits1(): 4
rCountEvenDigits2(): 4
```

(2) Test Case 2:
```
Enter the number:
23453
rCountEvenDigits1(): 2
rCountEvenDigits2(): 2
```

(3) Test Case 3:
```
Enter the number:
135
rCountEvenDigits1(): 0
rCountEvenDigits2(): 0
```

2. (**rAllEvenDigits**) The **recursive** function that returns either 1 or 0 according to whether or not all the digits of the positive integer argument number `num` are even. For example, if the argument `num` is 2468, then the function should return 1; and if the argument `num` is 2345, then 0 should be returned. Write the recursive function in two versions. The function `rAllEvenDigits1()` computes and returns the result. The function `rAllEvenDigits2()` computes and returns the result through the parameter `result` using call by reference. The function prototypes are given below:

```
int rAllEvenDigits1(int num);
void rAllEvenDigits2(int num, int *result);
```

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

```
#include <stdio.h>
#define INIT_VALUE 999
int rAllEvenDigits1(int num);
void rAllEvenDigits2(int num, int *result);
int main()
{
   int number, result=INIT_VALUE;

   printf("Enter a number: \n");
   scanf("%d", &number);
   result = rAllEvenDigits1(number);
   if (result == 1)
      printf("rAllEvenDigits1(): yes\n");
   else if (result == 0)
      printf("rAllEvenDigits1(): no\n");
   else
      printf("rAllevenDigits1(): error\n");
   result=INIT_VALUE;
   rAllEvenDigits2(number, &result);
   if (result == 1)
      printf("rAllEvenDigits2(): yes\n");
   else if (result == 0)
      printf("rAllEvenDigits2(): no\n");
   else
      printf("rAllevenDigits2(): error\n");
   return 0;
}
int rAllEvenDigits1(int num)
{
   /* Write your code here */
}
void rAllEvenDigits2(int num, int *result)
{
   /* Write your code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter a number:
5
rAllEvenDigits1(): no
rAllEvenDigits2(): no
```

(2) Test Case 2:

```
Enter a number:
2468
rAllEvenDigits1(): yes
rAllEvenDigits2(): yes
```

(3) Test Case 3:

```
Enter a number:
2345
rAllEvenDigits1(): no
rAllEvenDigits2(): no
```

(4) Test Case 4:

```
Enter a number:
```

```
280
rAllEvenDigits1(): yes
rAllEvenDigits2(): yes
```

3. **(rStrLen)** The **recursive** function that accepts a character string *s* as parameter, and returns the length of the string. For example, if *s* is "abcde", then the function will return 5. The function prototype is given as follows:

```
int rStrLen(char *s);
```

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

```
#include <stdio.h>
int rStrLen(char *s);
int main()
{
   char str[80];

   printf("Enter the string: \n");
   gets(str);
   printf("rStrLen(): %d\n", rStrLen(str));
   return 0;
}
int rStrLen(char *s)
{
   /* Write your program code here */
}
```

Some sample input and output sessions are given below:

(1) Test Case 1:

```
Enter the string:
abcde
rStrLen(): 5
```

(2) Test Case 2:

```
Enter the string:
abc de
rStrLen(): 6
```

(3) Test Case 3:

```
Enter the string:
a
rStrLen(): 1
```