To:
Tenouk

C LAB WORKSHEET 7a_1
Another C & C++ Repetition Construct: while Loop and do-while Loop 2

Items in this page:

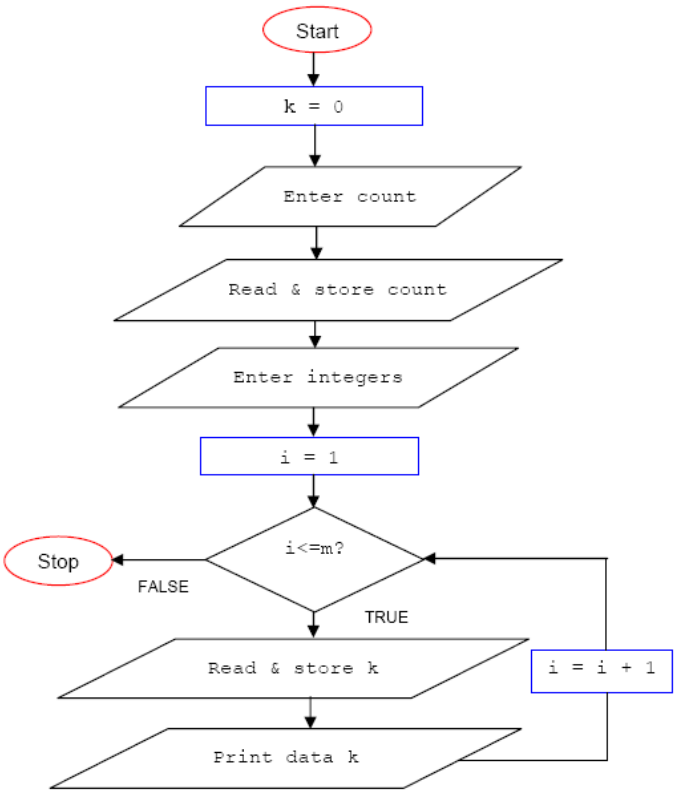
1. More for/while/do-while loop control activities, questions and answers.
2. The related tutorial reference for this worksheet are: C/C++ program control 1 and C/C++ program control 2.

```
#include <stdio.h>

int main()
{
   int k = 0, i, m;
   printf("Enter an integer as count: \n");
   scanf_s("%d", &m);
   printf("Enter the %d integers: \n", m);
   for( i = 1; i <= m; i = i + 1)
   {
     scanf_s("%d", &k);
     printf("The data is: %d \n", k);
   }
   return 0;
}
```

a. Was there any difference between the outputs of this experiment and the previous one?
b. Which of these two experiments preserved the number of times the loop was performed? That is, in which experiment did the value of m remain the same?
c. If the <= were changed to !=, then how many times would the loop have been executed? i would have gone up to what value?
d. Draw a flowchart for this program.

```
Enter an integer as count:
4
Enter the 4 integers:
1 2 3 4 5
The data is: 1
The data is: 2
The data is: 3
The data is: 4
Press any key to continue . . .
```

a. No difference.
b. This experiment because it uses index i for iteration.
c. 3 times. i = 3. i = i + 1 = 3 + 1 = 4.
d. See below




```
#include <stdio.h>

int main()
{
   int n, m;
   printf("Enter an integer: \n");
   scanf_s("%d", &m);
   printf("Enter another integer: \n");
   scanf_s("%d", &n);
   for( ; m < n;)
   {
     printf("The data is: %d \n", m);
     m = n;
     scanf_s("%d", &n);
   }
   return 0;
}
// try these input pair: 3, 3 then 3, 4 and finally 4, 3
```

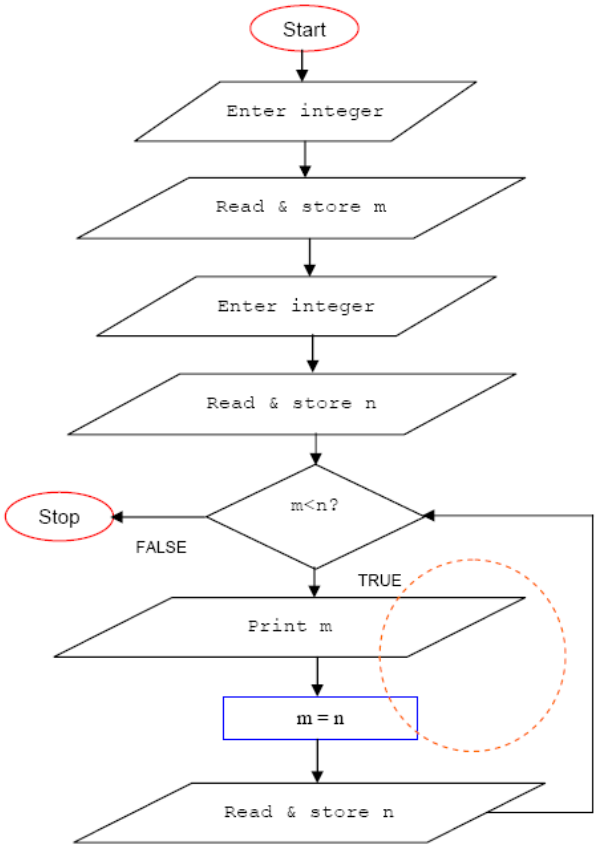
```
Enter an integer:
3
Enter another integer:
3
Press any key to continue . . .
```

```
Enter an integer:
3
Enter another integer:
4
The data is: 3
4
Press any key to continue . . .
```

```
Enter an integer:
4
Enter another integer:
3
Press any key to continue . . .
```

a. The first time the for loop is encountered, which variable was read in before the other, m or n?
b. Once inside the loop, there are two numbers available: the one that was read in earlier (outside for loop scanf_s()) and the one that was read in later (inside the for loop scanf_s ()). Inside the loop, which one is printed, the earlier one or the later one?
c. Inside the loop, does the earlier number now become the later one, or does the later one now become the earlier one?
d. Inside the loop, which one does the scanf_s() read, the earlier one or the later one?
e. After the scanf_s(), is m the earlier number or the later number?
f. Without using the names of the variables but instead the terms earlier number and later number, state when the loop continues and when it terminates.
g. Draw a flowchart and/or tracechart.

a. m and then n.
b. The earlier, that is m. The for loop only executed when m < n, that is the earlier value less than the later value else the for loop is skipped. That is why for the 3, 3 and 4, 3 input pairs, the for loop is skipped.
c. Earlier becomes the later, that is m becomes n. Then both will have same value.
d. The later, that is n.
e. Earlier number.
f. The loop continues only when the earlier number is less than the later number. Other value pairs will make the for loop being skipped.
g. See below.



It is very interesting for us to study the flow and behavior of the program. During the debugging process it is a repetitive process of editing, re-editing and commenting out the code. For example by adding suitable codes in the previous program as shown, we can see the program flow and behavior clearer as shown by a flowchart and a more details by tracechart.

```
Enter an integer:
3
Enter another integer:
4
The data m = 3 and n = 4
The data is: 3
The data m = 4 and n = 4
7
The data m = 4 and n = 7
The data is: 4
The data m = 7 and n = 7
11
The data m = 7 and n = 11
The data is: 7
The data m = 11 and n = 11
4
The data m = 11 and n = 4
Press any key to continue . . .
```

```
#include <stdio.h>

int main()
{
   int n, m;
   printf("Enter an integer: \n");
   scanf_s("%d", &m);
   printf("Enter another integer: \n");
   scanf_s("%d", &n);
   // see the current values
   printf("The data m = %d and n = %d\n", m, n);
   for( ; m < n;)
   {
      printf("The data is: %d \n", m);
      // assign n's value to m
      m = n;
      // see the current values
      printf("The data m = %d and n = %d\n", m, n);
      // waiting or reading next input
      scanf_s("%d", &n);
      // see the current values
      printf("The data m = %d and n = %d\n", m, n);
   }
   return 0;
}
```

Now change the conditional operator as shown below. Can you determine what will be printed?

```
Enter an integer:
3
Enter another integer:
3
Press any key to continue . . .
```

```
#include <stdio.h>

int main()
{
   int n, m;
   printf("Enter an integer: \n");
   scanf_s("%d", &m);
   printf("Enter another integer: \n");
```

```
    scanf_s("%d", &n);
    for( ; m != n; )
    {
      printf("The data is: %d \n", m);
      m = n;
      scanf_s("%d", &n);
    }
    return 0;
}
```

a. State what numbers are printed and when the loop terminates?

```
Enter an integer:
3
Enter another integer:
4
The data is: 3
3
The data is: 4
4
The data is: 3
5
The data is: 4
7
The data is: 5
20
The data is: 7
100
The data is: 20
_
```

a. Numbers are printed and the for loop been executed when the condition of m != n is TRUE. When we input number pairs that less than or greater than, the for loops is TRUE forever because the scanf_s() will read new different n value. Numbers printed are interchangeably between n and m. You need to terminate the program execution manually (Ctrl + C). If m == n, the for loop will terminate because m != n is FALSE.

```
#include <stdio.h>

int main()
{
    int k = 0, i;
    printf("Enter integers: \n");
    scanf_s("%d", &k);
    for(i = 0; k != 11; i = i + 1)
    {
      printf("The data is: %d \n", k);
      scanf_s("%d", &k);
    }
    printf("\n%d\n", i);
    return 0;
}
```

a. What does i represent when it is printed **after** the loop? And why?

```
Enter integers:
3 7 8 11 14 10 9 9 6
The data is: 3
The data is: 7
The data is: 8

3
Press any key to continue . . .
```

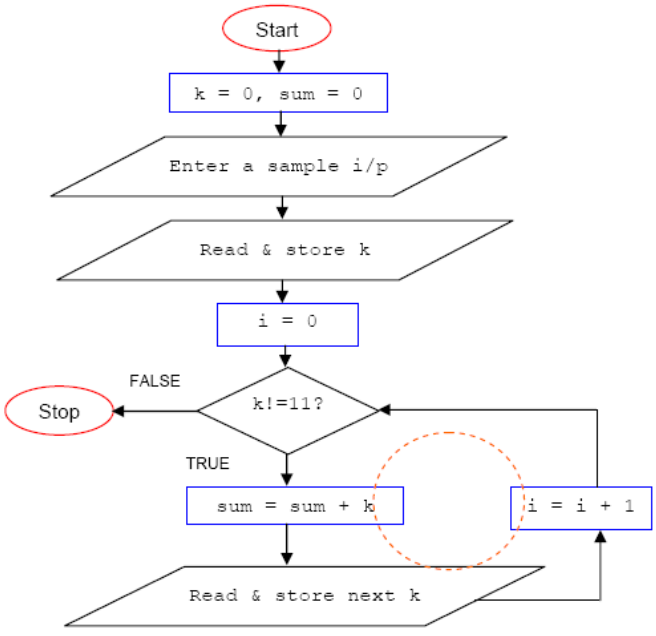
a. i represents an index of iteration. For three values, 3, 7 and 8, the index of iterations are 0, 1 and 2. So, the final value of i = i + 1 = 2 + 1 = 3.

```
#include <stdio.h>

int main()
{
    int k = 0, i, sum = 0;
    printf("Enter a sample input: \n");
    scanf_s("%d", &k);
    for(i = 0; k != 11; i = i + 1)
    {
      sum = sum + k;
      scanf_s("%d", &k);
    }
    printf("sum = %d, i = %d, ((sum*1.5)/i) = %.2f\n", sum, i, ((sum*1.5)/i));
    return 0;
}
```

a. After the loop, what do the three numbers represent?
b. Draw a flowchart and/or a tracechart.

```
Enter a sample input:
3 7 8 11 14 10 9 9 6
sum = 18, i = 3, ((sum*1.5)/i) = 9.00
Press any key to continue . . .
```

a. The three numbers represent the first three inputs.
b. See below.



Show the output for the following programs and from the output, study the program behaviors.

```c
#include <stdio.h>

int main()
{
   int k = 0, i, sum1 = 0, sum2 = 0;
   printf("Enter the sample input: \n");
   for(i=1; i<=2; i=i+1)
   {
    scanf_s("%d", &k, 1);
    sum1 = sum1 + k;
    scanf_s("%d", &k, 1);
    sum2 = sum2 + k;
   }
   printf("sum1 = %d, sum2 = %d\n", sum1, sum2);
   return 0;
}
// sample input: 2 40 30 90 10 40 on one line or separate line
```

```
Enter the sample input:
2 40 30 90 10 40
sum1 = 32, sum2 = 130
Press any key to continue . . .
```

```c
#include <stdio.h>

int main()
{
   int k = 0, sum = 0;
   printf("Enter the sample input: \n");
   scanf_s("%d", &k, 1);
   for(; k != 10;)
   {
    sum = sum - k;
    scanf_s("%d", &k, 1);
    printf("sum = %d ", sum);
   }
   printf("\nfinal sum = %d\n", sum);
   return 0;
}
// sample input: 2 40 30 90 10 40 on one line or separate line
```

```
Enter the sample input:
2 40 30 90 10 40
sum = -2 sum = -42 sum = -72 sum = -162
final sum = -162
Press any key to continue . . .
```

Write C code snippet solutions for the following problem statements. Before writing and testing the codes you might want to create a pseudocode and a flowchart and/or tracechart.

Ask the user how many numbers to be added. Read in that many numbers and print their sum at the end.

```c
#include <stdio.h>

int main()
{
   int i, k, j = 0;
   double sum = 0.0;
   // prompt the number of count
   printf("Enter number count to be summed up: \n");
   // read and store the count
   scanf_s("%d", &k, 1);
   // prompt for the integers
   printf("Enter those integers: \n");
   for(i = 1; i <=k; i++)
   {
      // start reading & storing the first input and so on
      scanf_s("%d", &j, 1);
      // sum the input
      sum = sum + j;
      // check, i <= k?, if TRUE, repeat, else
      // stop or exit the loop
   }
   // print the final sum
   printf("\nFinal sum = %.2f\n", sum);
   return 0;
}
```

```
Enter number count to be summed up:
4
Enter those integers:
7 3 8 1

Final sum = 19.00
Press any key to continue . . .
```

Read in grades for two quizzes (floats) on each line (for each student) until a -1 is entered for the first quiz. Print the average of each quiz at the end.

```c
#include <stdio.h>

int main()
{
   int i;
   float mark = 0.0, sum = 0.0;
   printf("Enter the 1st & 2nd marks, 1st mark -1 to stop: \n");
   scanf_s("%f", &mark);
   for(i = 0; mark != -1.0; i = i + 1)
   {
      sum = sum + mark;
      scanf_s("%f", &mark);
   }
   printf("Total mark = %.2f, number of mark = %d,\n Average: %  .2f\n", sum, i,
(sum/i));
   return 0;
}
```

```
Enter the 1st & 2nd marks, 1st mark -1 to stop:
50.5 20.7
88.5 57.6
40.5 67.5
-1
Total mark = 325.30, number of mark = 6,
 Average: 54.22
Press any key to continue . . .
```

Read in grades with a value between 0 and 100 until a negative grade is read in. Stop the loop once a negative grade is read in. At the end print the average of all the other grades except for the negative one.
For your own observation: Change the terminal condition in the for statement: (mark > 0) & (mark <100) to the following, rebuild and re-run your program and see the effect, error or warning.

a. 0<mark<100.
b. 0<=mark<=100.
c. (mark >0) | (mark < 100).
d. (mark >=0) | (mark <= 100).
e. 100>mark>0.
f. (100 > mark) & (0 < mark).

```
#include <stdio.h>

int main()
{
   int i;
   float mark = 0.0, sum = 0.0;
   printf("Enter marks, -ve to stop: \n");
   scanf_s("%f", &mark);
   for(i = 0; (mark > 0) & (mark <100); i = i + 1)
   {
      sum = sum + mark;
      scanf_s("%f", &mark);
   }
   printf("Total mark = %.2f, number of mark = %d,\n Average: %.2f\n", sum, i,
(sum/i));
   return 0;
}
```

```
Enter marks, -ve to stop:
45.50
60.45
70.60
80.5
-1
Total mark = 257.05, number of mark = 4,
 Average: 64.26
Press any key to continue . . .
```

Keep reading in characters until two consecutive characters are equal. Then print the total number of characters except for the last two.

```
#include <stdio.h>
int main()
{
   char n, m;
   int i;
   // read char & store in m
   printf("Enter a character, same char to terminate: ");
   scanf_s(" %c", &m);
   // then read char & store in n
   printf("Enter another character, same char to terminate: ");
   scanf_s(" %c", &n);
   // if m!=n, that is different char, execute for loop..
   // else just go to the next statement
   // after the for loop body
   // i used to calculate the total character minus the
   // last similar character
   for(i = 0; m != n; i=i+1)
   {
      printf("More character, same char to terminate: ");
      // assign n's value to m, so m will hold n's value
      m = n;
      // then read next char & store in n, so
      // n will hold new value, different with m
      // Then both will hold fresh values...
      // though the first n and m are outside the for loop...
      scanf_s(" %c", &n);
   }
   printf("Total character: %d\n", i);
   return 0;
}
```

```
Enter a character, same char to terminate: a
Enter another character, same char to terminate: a
Total character: 0
Press any key to continue . . .
```

```
Enter a character, same char to terminate: a
Enter another character, same char to terminate: b
More character, same char to terminate: c
More character, same char to terminate: d
More character, same char to terminate: e
More character, same char to terminate: e
Total character: 4
Press any key to continue . . .
```

More Practice

Run the following programs and answer the questions.

```
#include <stdio.h>

int main()
{
    int i, k, total;
    total = 0;
    printf("Enter 3 integers: ");
    for(i=1; i <=3; i=i+1)
    {
        // 1st iteration, read and store the first input
        // next iteration, read and store next input
        scanf_s("%d", &k);
        // sum up for every iteration
        total = total + k;
        // increment i & check the terminal condition
    }
    printf("Total = %d\n", total);
    return 0;
}
```

a. How many times is this loop **executed**? Does this answer depend on what the **data** was?

```
Enter 3 integers: 10 12 31
Total = 53
Press any key to continue . . .
```

a. 3 times. No. though you enter more than 3 inputs, the loop been executed 3 times only.
b. 3 numbers.
c. 1. 40 2. 90 3. 110 4. 110

You can see this effect by adding the following code after total = total + k;. Re-build and re-run the program.
printf("total hold: %d value at this moment.\n", total);

```
Enter 3 integers: 40 50 20
total hold: 40 value.
total hold: 90 value.
total hold: 110 value.
Total = 110
Press any key to continue . . .
```

b. How many numbers will the loop **read**?
c. If 40, 50 and 20 were read in, what would be the value of total:

1. At the end of the first iteration?
2. At the end of the second iteration?
3. At the end of the third iteration?
4. After the loop when it is printed?

d. If 90, 10 and 30 were read in, then what would be printed?

```
#include <stdio.h>

int main()
{
    int i, k, total;
    total = 0;
    printf("Enter an integer:\n");
    scanf_s("%d", &k, 1);
    printf("More integers:\n");
    for(i=1; i <=3; i=i+1)
    {
        total = total + k;
        scanf_s("%d", &k, 1);
    }
    printf("Total = %d\n", total);
    return 0;
}
```

a. How many times is the loop performed?
b. How many times is each of the statements in the loop performed?
c. How many times is scanf_s() executed before, during and after the loop?
d. How many data items are read in?
e. If the data items are 40, 20, 50 and 10, then what is the value of total after the first time, second time and the third times through the loop?
f. Is the first data item added into the loop? What about the last one?
g. Draw a flowchart and/or tracechart for this program. The data items are 40, 20, 50 and 10. For the tracechart, make columns for k, i, total and i<3?

d. 130.

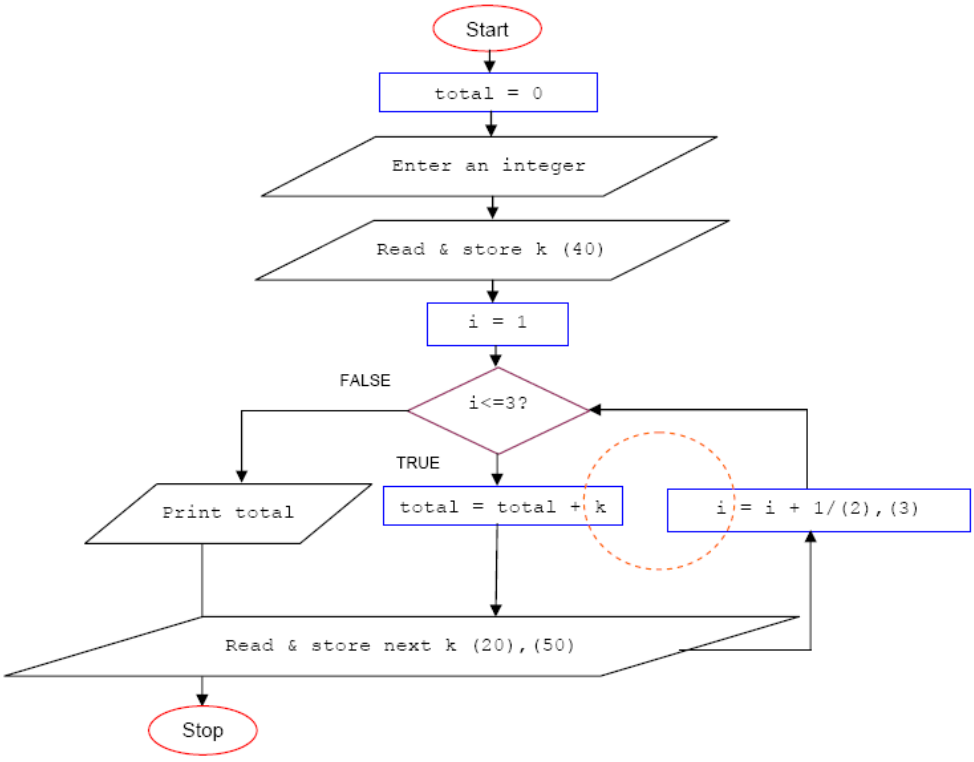
```
Enter an integer:
10
More integers:
20
30
40
Total = 60
Press any key to continue . . .
```

a. 3 times.
b. 3 times.
c. Before the loop is once. During the loop is 2 times. Not executed after the loop.
d. 3 data items. In this case: 10, 20 and 30.
e. 40, 60 and 110. We put the following line of code just after total = total + k; code.

printf("total holds before: %d\n", total);
The output is shown below.

```
Enter an integer:
40
More integers:
total holds before: 40
20 50 10
total holds before: 60
total holds before: 110
Final total = 110
Press any key to continue . . .
```

f. The first data item was added but the last one was not. This is because the i index already equal to 3.
g. See below




```
#include <stdio.h>

int main()
{
    int i, k, total;
    total = 0;
    printf("Enter an integer:\n");
    scanf_s("%d", &k, 1);
    printf("More integers:\n");
    for(i=1; i <=3; i=i+1)
    {
        scanf_s("%d", &k, 1);
        total = total + k;
    }
    printf("Total = %d\n", total);
    return 0;
}
// try the sample input: 40, 20, 50 and 10
```

```
Enter an integer:
40
More integers:
20
50 10
Total = 80
Press any key to continue . . .
```

a. Before is once. During the loop is three and after the loop is none.
b. 20.
c. No.
d. Yes.
e. 80. To see the behavior of this program add the following code just after the total = total + k;

printf("Current total = %d\n", total);

a. How many times is the scanf_s() done before, during and after the loop?
b. After the first iteration of the loop, what is the value of total?
c. Was the first data item added into total?
d. Was the last data item added into total?
e. What will be the final printout?

```
#include <stdio.h>

int main()
{
    int i, k, total;
    total = 0;
    printf("Enter an integer:\n");
    scanf_s("%d", &k, 1);
    printf("More integers:\n");
    for(i=1; i <=3; i=i+1)
        scanf_s("%d", &k, 1);   // statement 1
            total = total + k;      // statement 2
    printf("Total = %d\n", total);
    return 0;
}
// sample input: 40, 20, 50 and 10
```

a. Which statement(s) are in the body of the loop?
b. Which statement is indented incorrectly?
c. Will 40 be added to total?
d. Which numbers are added to total?
e. Out of the numbers that were added to the total, which ones were added inside the loop and after the loop?

Using the following for loop:

```
for(i=1; i <= count; i = i + 1)
```

Write a program that will first read the variable count as a data item, read in that many more data items and print their total. For example, if the data were 3, 40, 20, 50, and 10; count would become 3 and the sum of the next 3 data items would be calculated as 110.

a. Was the first data item that was read added to total?
b. What was the purpose of the first data item?
c. If the first data item were a 7, how many numbers would the program attempt to add?

```
#include <stdio.h>

int main()
{
    int i, k;
    printf("Enter an integer:\n");
    scanf_s("%d", &k, 1);
    for(i=1; k !=0; i=i+1)
        scanf_s("%d", &k, 1);
    printf("i = %d\n", i);
    return 0;
}
// sample input: 4, 2, 9, 11, 4 and 0
```

a. What is the output if the input were 4, 2, 9, 11, 4 and 0?
b. Rewrite the program so that the code will not print the count of data items that were read in, but instead will print out the average of all numbers except the 0. Use sum as the variable to add the numbers.
c. Rewrite the program in (b) so that the loop stops when the new number is less than the previous one. That is, if the data items were 30, 70, 85, 80, then the loop should stop at 80 but print out the average of all the numbers up to 85, the one before the last one. You will need two variables, call them last and curr. You will also need two scanf() or scanf_s () before the loop. Clue: as long as curr is greater than last, the loop continues.
d. Draw a flowchart and/or tracechart for (c).

```
Enter an integer:
40
More integers:
20 50 10
Current total = 20
Current total = 70
Current total = 80
Final total = 80
Press any key to continue .
```

e. 80.

```
Enter an integer:
40
More integers:
20 50 10
Total = 10
Press any key to continue . . .
```

a. scanf_s("%d", &k, 1);, Statement 1
b. total = total + k;, Statement 2
c. No.
d. The last one, 10.
e. None were added inside the loop. 10 was added outside the loop.

```
Enter a counter: 4
Enter integers: 40 20 50 10
Total = 120
Press any key to continue . . .
```

a. No.
b. As a counter used as terminal condition in the for loop statement.
c. 7 numbers as shown in the following output.

```
Enter a counter: 7
Enter integers: 10 30 10 20 10 50 20 70
Total = 150
Press any key to continue . . .
```

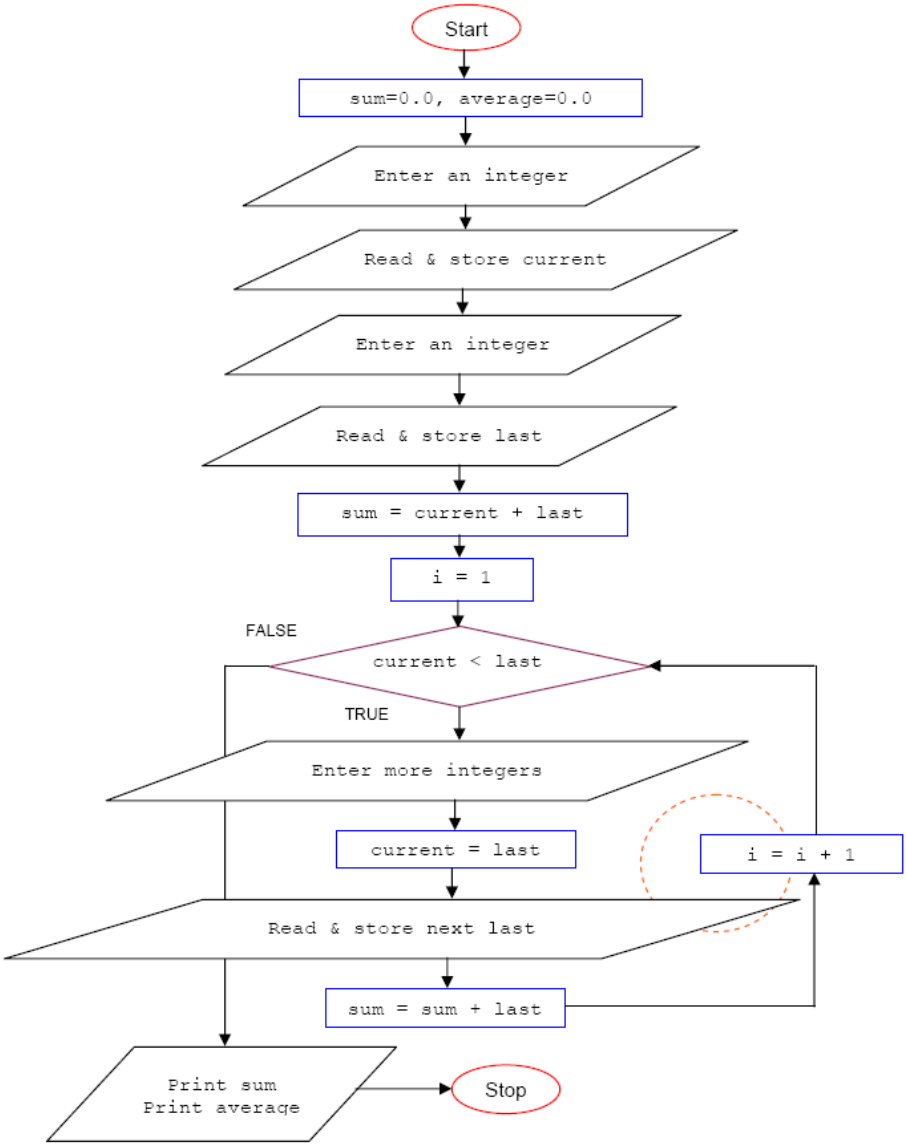
```
Enter integers:
4 2 9 11 4 0
i = 6
Press any key to continue . . .
```

a. 6.
b.
```
#include <stdio.h>

int main()
{int i, k, sum = 0;
printf("Enter integers:\n");
scanf_s("%d", &k, 1);
for(i=1; k !=0; i=i+1)
{
   sum = sum + k;
   scanf_s("%d", &k, 1);
}
printf("The integers sum = %d\n", sum);
return 0;
}
```

```
Enter integers:
10 10 10 20 30 10 50 0
The integers sum = 140
Press any key to continue .
```

c.
```
#include <stdio.h>

int main()
{
   float i, last, current, sum = 0.0, average = 0.0;
   printf("Enter an integer: ");
   scanf_s("%f", &current);
   printf("Enter another integer: ");
   scanf_s("%f", &last);
   sum = current + last;
   for(i = 1; current < last; i=i+1)
   {
       printf("More integer: ");
      // the current become the last value
      current = last;
      // read & store a fresh last value
      scanf_s("%f", &last);
      sum = sum + last;
   }
   printf("Sum of integers: %.2f and average = %.2f\n", (sum - last),
(sum-last)/i);
   return 0;
}
```

```
Enter an integer: 2
Enter another integer: 3
More integer: 5
More integer: 7
More integer: 10
More integer: 8
Sum of integers: 27.00 and average = 5.40
Press any key to continue . . .
```



| Main |< C & C++ while & do-while loop 1 | C & C++ if, if-else and switch-case-break 1 >|
Site Index | Download |

The C Repetition for, while and do-while: Part 1 | Part 2 | Part 3 | Part 4