C LAB WORKSHEET 8a
C & C++ Selection: C/C++ if and if-else Part 3

Items in this page:
1. The C & C++ conditional statement, a selection and flowcharts.
2. The if, if-else construct, variations and flowcharts.
3. Activities, questions and answers.
4. Tutorial reference that should be used together with this worksheet are: C & C++ program control 1 and C & C++ program control 2.

9. The following experiment should give the same result as the previous one. Only the logic has been rearranged. Complete the flowchart and the program so that the output is same as before.

```c
#include <stdio.h>

int main(void)
{
    int i, k, a = 0, b = 0, lower = 0;
    printf("Enter the sample input line by line:\n");
    for(i = 1; i <= 7; i = i + 1)
    {
        scanf("%d", &k, 1);
        if(k < 90)
        {
            if(k < 80)
                _______________
            else
                _______________
        }
        else
        _______________
    }
    printf("A's = %d	B's = %d	Lower = %d\n", a, b, lower);
    return 0;
}
```

10. Next, let us test the conditions for all five grades, namely A, B, C, D and F. In the blank space, for each grade, place the appropriate statement something like the following:

```c
#include <stdio.h>

int main(void)
{
    int i, k;
    printf("Enter the sample input line by line:\n");
    for(i = 1; i <= 7; i = i + 1)
    {
        scanf("%d", &k, 1);
        if(k < 90)
        {
            if(k < 90)
                _______________
            if(k < 80)
                _______________
            if(k < 70)
                _______________
            else
                _______________
        }
        else
        _______________
    }
    printf("A's = %d\tB's = %d\tC's = %d\tD's = %d\tF's = %d\n", a, b, c, d, f);
    return 0;
}
```
```c
if(k < 80)
    if(k < 70)
        if(k < 60)
            
        else
            
    else
        
else
    
else
    
else
    
}
return 0;
```
e. Grades that end up getting a D must go through how many decision diamonds?

Ans:

a. B, C, D and F.
b. C, D and F.
c. On the True side are D and F. On the False side, C was selected.
d. 2 decision diamonds.
e. 4 decision diamonds.

11. The following experiment performs the same steps as in the previous one. However, since the logic is rearranged, the `printf()` will need to be placed at different locations. Complete the code and the flowchart.

```c
#include <stdio.h>

int main(void)
{
    int i, k;
    printf("Enter the sample input line by line:\n");
    for(i = 1; i <= 7; i = i + 1)
    {
        scanf_s("%d", &k, 1);
        if(k > 69)
            if(k > 89)
                _______________________________
            else if(k > 79)
                _______________________________
            else
                _______________________________
        else if(k > 59)
            _______________________________
        else
            _______________________________
    }
    return 0;
}
```

```c
if(k > 69)
    if(k > 89)
        printf("Grade A.\n");
    else if(k > 79)
        printf("Grade B.\n");
    else
        printf("Grade C.\n");
else if(k > 59)
    printf("Grade D.\n");
else
    printf("Grade F.\n");
```

The following is a completed flowchart for the previous question.
a. F grades will go through two conditions: k > 69?, which would be false and k > 59?, which also would be false.
   How many conditions that D grades go through?

b. How many conditions that C grades go through?

c. How many conditions that A grades go through?

d. If k > 79? were changed to k <= 80?, then what changes would be necessary in the flowchart?

e. When a grade that is read into the variable k enters this set of nested if’s, it has a choice of going through
   how many different paths?

f. The control of execution may take how many different paths at any one time?

Ans:

a. Also 2 conditions same as F but both are True.
   b. 3 conditions.
   c. 2 conditions.
   d. If k > 79? were changed to k <= 80?, then what changes would be necessary in the flowchart?
   e. When a grade that is read into the variable k enters this set of nested if’s, it has a choice of going through
      how many different paths?
   f. The control of execution may take how many different paths at any one time?

12. The following experiment performs the same steps as in the last two previous experiments. However, it count the number of grades in each
category instead of printing them. Complete the code and the flowchart.

```c
#include <stdio.h>

int main(void)
{
    int i, k, a=0, b=0, c=0, d=0, f=0;
    printf("Enter the sample input line by line:\n");
    for(i = 1; i <= 7; i = i + 1)
    {
        if(k <= 59) f = f + 1;
        else if(k <= 89)
        {
            if(k <= 69) d = d + 1;
            else if(k <= 79) c = c + 1;
            else b = b + 1;
        }
        else a = a + 1;
    }
    printf("A's = %d\t", a);
    printf("B's = %d\t", b);
    printf("C's = %d\t", c);
    printf("D's = %d\t", d);
    printf("F's = %d\t", f);
    return 0;
}
```

if(k <= 59) f = f + 1;
else if(k <= 89)
    if(k <= 69) d = d + 1;
    else if(k <= 79) c = c + 1;
    else b = b + 1;
else a = a + 1;
For each of these questions, choose from among the grades of A, B, C, D and F.

a. Which grades(s) are selected on the true side of \( k \leq 59 \)?
   a. F.
   b. A, B, C and D.
   c. B, C and D.
   d. A.
   e. D.
   f. B and C.
   g. C.
   h. B.

b. Which grades(s) are selected on the false side of \( k \leq 59 \)?
   b. A, B, C and D.

For each of these questions, choose from among the grades of A, B, C, D and F.

c. Which grades(s) are selected on the true side of \( k \leq 89 \)?
   d. Which grades(s) are selected on the false side of \( k \leq 89 \)?

f. Which grades(s) are selected on the true side of \( k \leq 69 \)?
   g. Which grades(s) are selected on the true side of \( k \leq 79 \)?

h. Which grades(s) are selected on the false side of \( k \leq 79 \)?
13. Run the following program and key in the following sample input (without the commas): 16, 21, 17, 43, 7, 52, -1. The program will determine the smallest number entered.

```c
#include <stdio.h>
int main(void)
{
    int k, smallest;
    printf("Enter integers, when done enter a ");
    printf("negative number
");
    scanf_s("%d", &k, 1);
    // assign the first number to smallest variable
    smallest = k;
    // iterate while k >= 0
    for( ; k >= 0; )
    {
        // if the entered number is < smallest
        if(k < smallest)
            // then assign the number to smallest variable...
            smallest = k;
        // read the next input...
        scanf_s("%d", &k, 1);
    }
    // print the smallest number...
    printf("The smallest number is %d\n", smallest);
    return 0;
}
```

a. Draw a tracechart for this experiment (left to you!).
b. What was the first value of the variable smallest?
c. The first time that the condition in the if statement was encountered, what were the values of k and smallest?
d. The second time that the if condition was tested, what were the values of k and smallest?
e. The third time that the if condition was tested, what were the values of k and smallest?
f. During the loop, the value of smallest was changed. What were the different values of smallest?
g. Does an if statement require a corresponding else statement? Why?
h. Is the scanf_s() executed inside the loop when the k < smallest is true or false, or irrespective of it?
i. If the data entered were 11, 22, 13, 19, 16, -1, how many times would smallest be changed?

To see the flow of this program clearer and used for troubleshooting you can add several line of codes as shown below.

```c
#include <stdio.h>
int main(void)
{
    int k, smallest;
    printf("Enter integers, when done enter a ");
    printf("negative number
");
    scanf_s("%d", &k, 1);
    // assign the first number to smallest variable
    smallest = k;
    printf("smallest = %d, k = %d at pos1.\n", smallest, k);
    // iterate while k >= 0
    for( ; k >= 0; )
    {
        // if the entered number is < smallest
        printf("smallest = %d, k = %d at pos2.\n", smallest, k);
        if(k < smallest)
            // then assign the number to smallest variable...
            smallest = k;
        printf("smallest = %d, k = %d at pos3.\n", smallest, k);
        // read the next input...
        scanf_s("%d", &k, 1);
        printf("smallest = %d, k = %d at pos4.\n", smallest, k);
    }
    // print the smallest number...
    printf("The smallest number is %d\n", smallest);
    return 0;
}
```

a. Left for your assignment.
b. 16
c. k = 16, smallest = 16.
d. k = 21, smallest = 16.
e. k = 17, smallest = 16.
f. 16 and 7.
g. Not really. It can be standalone in testing just a condition.
h. From the flowchart we can see that the scanf_s() executed inside the loop irrespective of k < smallest is True or False.
i. 0 time. It is already a smallest number entered as the first input.

To see the flow of this program clearer and used for troubleshooting you can add several line of codes as shown below.

```c
#include <stdio.h>
int main(void)
{
    int k, smallest;
    printf("Enter integers, when done enter a ");
    printf("negative number
");
    scanf_s("%d", &k, 1);
    // assign the first number to smallest variable
    smallest = k;
    printf("smallest = %d, k = %d at pos1.\n", smallest, k);
    // iterate while k >= 0
    for( ; k >= 0; )
    {
        // if the entered number is < smallest
        printf("smallest = %d, k = %d at pos2.\n", smallest, k);
        if(k < smallest)
            // then assign the number to smallest variable...
            smallest = k;
        printf("smallest = %d, k = %d at pos3.\n", smallest, k);
        // read the next input...
        scanf_s("%d", &k, 1);
        printf("smallest = %d, k = %d at pos4.\n", smallest, k);
    }
    // print the smallest number...
    printf("The smallest number is %d\n", smallest);
    return 0;
}
```

a. Sample inputs: 16, 21, 17, 43, 7, 52, -1 and 11, 22, 13, 19, 16, -1