SHEET 2 SOLUTIONS  
SHEHAB MAHMOUD SALAH | 2100320

**GENERAL NOTE**: to **copy** code **from** this PDF document, **copy** **each** **block** **of** **code** separately to **not** lose the code’s formatting, **happy compiling** 😄

1. Review Questions

1. A) Wrong syntax, the correct statement would be:

**if** (quarters > 0){  
cout << quarters << “quarters”;  
}

B) Wrong syntax, the correct statement would be:

**if** (1+x > pow(x , sqrt(2))) {  
y = y+x;  
}

C) The correct code would be:

**if** (x == 1) {  
 y++;  
}  
**else if** (x == 2) {  
 y = y + 2;  
}

D) The correct code would be:

**if** (x == 0 && y == 0) {  
 cout << Point(0,0);  
 }

E) The correct code would be:

**if** (1 <= x && x <= 10) {  
cout << “Enter y: “;  
cin >> y;  
}

F) The correct code would be:

**if** (s != "nick" && s != "penn" && s != "dime" && s != "quar") cout << "Input error!"*;*

*Explanation:* The logical operator that should be used is the AND operator, because using the OR operator means the condition will always be true regardless of what “s” is, and you will always get “Input error!” as an output.

G) **if** (input == “N” || “**NO**”) **return** 0;

H) cin >> x; **if**(!cin.fail()) y = y + x;

I) language = “English”;  
**if** (country == “USA”) {  
 **if** (state == “PR”) language = “Spanish”;  
 }  
**else** **if** (country = “China”) language = ”Chinese”;

2. A. Tom, Dick - **Dick comes first**   
B. Tom, Tomato - **Tom comes first**   
C. Car manufacturer, carburetor - **Car manufacturer comes first**   
D. Harry, hairy - **Harry comes first**   
E. C++, Car - **C++ comes first**   
F. Car, Carl - **Car comes first**

3.   
A) **15.9** The stream will read 2 characters (15) and stop at the decimal point. **The stream will not fail.**B) **15 9** The stream will read 2 characters (15) and stop at the space. **The stream will not fail.**   
C) **+159** The stream will read 4 characters (+159) and stop at the end of line. **The stream will not fail.**D) **-15A9** The stream will read 3 characters (-15) and stop at A. **The stream will not fail.**   
E) **Fifteen** **The stream will fail immediately as F is not a digit or a sign**.   
F) **-Fifteen** **The stream will fail immediately as F is not a digit after -.**   
G) **+ 15** **The stream will fail immediately as there is no digit after +.**   
H) **1.5E3** The stream will read 1 character (1) and stop at the decimal point. **The stream will not fail.**   
I) **+1+5** The stream will read 2 characters (+1) and stop at +. **The stream will not fail.**

4. An if/else/else statement is a way of branching the execution flow based on different conditions. For example:

if (x > 0) {

cout << "x is positive" << endl;

}

else if (x < 0) {

cout << "x is negative" << endl;

}

else {

cout << "x is zero" << endl;

}

This code will print different messages depending on the value of x. The else if clause is executed only if the previous if condition is false. The else clause is executed only if all the previous conditions are false.

A nested if statement is an if statement that is inside another if statement. For example:

if (y > 10) {

cout << "y is greater than 10" << endl;

if (y % 2 == 0) {

cout << "y is also even" << endl;

}

}

This code will print two messages only if y is greater than 10 and even. The inner if condition is evaluated only if the outer if condition is true.

Nested if statements can be equivalent to single if statements with a combined condition using logical operators like && (and) or || (or). For example, the above code can be rewritten as:

if (y > 10 && y % 2 == 0) {

cout << "y is greater than 10 and even" << endl;

}

5. Here is an example of an if/else/else statement where the order of the tests does not matter:

**if** (x > 0 **&&** y > 0){   
 cout << x + y << endl;  
**} else** {  
 cout << x – y << endl;  
}

In this example, the if statement will always print the sum of x and y, regardless of the order of the tests.

Here is an example of an if/else/else statement where the order of the tests matters:  
  
if (x > y) {  
 cout << x << endl;

} else {  
 cout << y << endl;

}

In this example, the if statement will only print x if x is greater than y, and will only print y if y is greater than x. The order of the tests matters because the if statement will only check the first test that is true.

6.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **p** | **q** | **r** | **(p && q) || !r** | **!(p && (q || !r))** |
| 0 | 0 | 0 | **1** | **1** |
| 0 | 0 | 1 | **0** | **1** |
| 0 | 1 | 0 | **1** | **1** |
| 0 | 1 | 1 | **0** | **1** |
| 1 | 0 | 0 | **1** | **0** |
| 1 | 0 | 1 | **0** | **0** |
| 1 | 1 | 0 | **1** | **0** |
| 1 | 1 | 1 | **1** | **0** |

7. To formulate the condition "**x is positive**", you can use the following code:

if (x > 0) {

// Code to be executed if x is positive

}

To formulate the condition "**x is zero or negative**", you can use the following code:

if (x <= 0) {

// Code to be executed if x is zero or negative

}

To formulate the condition "**x is at least 8**", you can use the following code:

if (x >= 8) {

// Code to be executed if x is at least 8

}

To formulate the condition "**x is less than 8**", you can use the following code:

if (x < 8) {

// Code to be executed if x is less than 8

}

To formulate the condition "**x and y are both zero**", you can use the following code:

if (x == 0 && y == 0) {

// Code to be executed if x and y are both zero

}

8. The difference between the two code segments is that the second one has an else if statement. **This means that if x is positive, s will be incremented, and if y is positive, s will also be incremented**. However, if x is negative, only y will be incremented.

In the first code segment, **s will only be incremented if x is positive**. If y is negative, s will remain at 0.

9. **An infinite loop is a sequence of instructions that repeats itself indefinitely**. This can be done by using a while loop or a for loop.

To terminate a program that executes an infinite loop, you can use the CTRL+C keyboard shortcut. This will send a SIGINT signal to the program, which will cause it to stop executing.

You can also use the Task Manager to terminate a program that executes an infinite loop. To do this, open the Task Manager and find the program that is executing the infinite loop. Then, right-click on the program and select "End Task".

10. A. The loop will execute 10 times, starting from i = 1 and ending at i = 10.

B. The loop will execute 9 times, starting from i = 0 and ending at i = 9.

C. The loop will execute 9 times, starting from i = 10 and ending at i = 1.

D. The loop will execute 21 times, starting from i = -10 and ending at i = 10.

E. The loop is an infinite loop.

F. The loop will execute 11 times, starting from i = -10 and ending at i = 10.

G. The loop will execute 7 times, starting from i = -10 and ending at i = 7.

11. int s{};

int i = 1;

while (i <= 10) {

s = s + i;

i++;

}

12.

int n;  
cin >> n;

double x = 0;

double s = 0;

while (s > 0.01)

{

s = 1.0 / (1 + n \* n);

n++;

x = x + s;

}

*The only difference is that the while loop checks the condition at the top of the loop, while the do/while loop checks the condition at the bottom of the loop. This means that the do/while loop will always execute the loop body at least once, even if the condition is false.*

13. **A. s =00, n = 00**

**B. s = 00, n = 00**

**C. s = 00, n = 00**

14.