

Answers to the 20 theory question on selection

Q1. Sequence(Sop), Selection(if), Iteration(for), Jump(break).

Q2. Simple- one condition, compound – 2 or more joined using logical operators.

Evaluation- &&- all the conditions should be true, || any one(or more) condition needs to be true. (Show truth table of && and ||.)

Q3. Working- In the absence of *else* Control goes to the next statement if the condition is false. Flowchart:

Syntax	Presence of ELSE	Absence of ELSE
if(Condition) Statement 1; [else Statement 2;]		
<i>Note- Statements need to be in {} only if they are more than 1.</i>		

Q4. Set of statements in {}, Block, when we have more than one statement inside if.

Q5.	<pre>if(a&gt;b) { Sop("XYZ"); }</pre>	<pre>if(a&gt;b) { if(a&gt;c)   Sop("xyz"); }</pre>	<pre>if(a&gt;b) { Sop("xyz"); } else if(b&gt;a) { Sop("pqr"); }</pre>	<pre>if(a&gt;b) { Sop("xyz"); } if(b&gt;a) { Sop("pqr"); }</pre>	<pre>if(a&gt;b) { Sop("xyz"); } else if(a&gt;b) { Sop("pqr"); } else if(a&gt;c) { Sop("abc"); }</pre>
	Single Line	Nested If	Else If	Multiple If	If-Else Ladder

Q6. An else with more than 2 ifs above it. By using {}.

Q7	?:	IF	?:
	An expression Single line Condition not in () Else not optional	A statement Can have many lines Condition must be in () Else optional	Advantage – Short.  Disadvantage – Complex in nested form.

Q8	Switch	If
	One variable Int or char only allowed Matching with constants Only equality checked case, default Faster	Can be more All types Can match with any type of values. All operators allowed else Slower

Q9. Syntax: condition? true-expression : false-expression

Example (of nested form) `g = a > b ? a > c ? a : c : b > c ? b : c ;`

Syntax of switch	Example of nested form
<pre>switch(expression) {   case value1: statements               [break;]   case value n: statements               [break;]   [default   : statements               [break;] }</pre>	<pre>switch(class) {   case 1: switch(section)           {             case 'A' : Sop("1-A"); break;             case 'B' : Sop("1-B"); break;           }           break;   case 2: switch(section)           {             case 'A' : Sop("2-A"); break;             case 'B' : Sop("2-B"); break;           }           break; }</pre>

Q10. When all the cases after the true case are executed – useful when we have a common body for more than one case – Can be prevented using break.

Q11. Relational - < <= > >= == !=, Logical: && || !

- & - checks all the conditions                      && - checks only till required.
- can operate on numbers (bitwise)                      cannot work on numbers
- | - True of any one or both true                      ^ - True of one and only one true.

Q12. Executed when no case is true.

Q13. (i) Character constants (ii) Unicode (iii) Character functions.

Q14.	AND – nested if		OR – else if	
	<pre>IF(A&gt;B &amp;&amp; A&gt;C) {   SOP("YES") }</pre>	<pre>IF(A&gt;B) {   IF(A&gt;C)   {     SOP("YES")   } }</pre>	<pre>IF(A&gt;B    A&gt;C) {   SOP("YES") }</pre>	<pre>IF(A&gt;B) {   SOP("YES") } ELSE IF(A&gt;C) {   SOP("YES") }</pre>

Q15. Switch statement to replace OR -

<pre>if(a==1) sop("one"); else if(a==2) sop("two"); else if(a==3) sop("three"); else sop("not 1 2 or 3");</pre>	<pre>switch(a) {     case 1: sop("one");             break;     case 2: sop("two");             break;     case 3: sop("three");             break;     default: sop("not 1 2 or 3"); }</pre>
---	---

Q16. Because NOT reverses a condition and reverses of all the relational operators are already available E.g. == reverse !=, < reverse >= etc.

Q17. ; (Semi colon).

Q18.	Multiple if →	All conditions tested	Compound conditions required	Last condition cannot be omitted.	Slower
	Nested If →				

Q19. An equation can also be given in switch. E.g. SWITCH( N%2)

```
{
}
```

Q20. With functions: if(Character.isLetterOrDigit(c)==false), if( ! Character.isLetterOrDigit(c))

Without functions: if( ! ( (a>=65 && a<=90) ||(a>=97 && a<=122) || (a>=48 && a<=57) ) )

END