9.2.1 Logic statements and decisions in an algorithm

Name: _____ Class: _____ Date: _____ **Total: 51 marks**

KEY WORDS logic statements • decision • selection • pseudocode • identifier

Objective

Learn to write the **logic statements** that an algorithm uses to make a **decision**. Sheet 9.2 wrote algorithms that run straight through; here the algorithm must choose. You write conditions with comparison operators and with `AND`, `OR` and `NOT`, work out whether they are TRUE or FALSE, and use one in a simple `IF`. The full rules for `IF` and `CASE` are on sheets 11.2 and 11.2.1.

You must be able to:

- write a logic statement for a condition that is given in words
- use the comparison operators, and join comparisons with `AND`, `OR` and `NOT`
- write the condition for a value **inside** a range and **outside** a range
- work out whether a logic statement is TRUE or FALSE for given values
- use a logic statement to define a decision in an algorithm

1 Worked examples

Study these first. Each one shows a **method** that a question later on this sheet needs.

Comparisons

- A **logic statement** is a condition that is either **TRUE** or **FALSE**.
- The simplest one **compares** two values:

operator	meaning	with `Age` = 16
`=`	equal to	`Age = 16` is TRUE
`<>`	not equal to	`Age <> 16` is FALSE
`<`	less than	`Age < 16` is FALSE
`>`	greater than	`Age > 12` is TRUE
`<=`	less than or equal to ("at most")	`Age <= 16` is TRUE
`>=`	greater than or equal to ("at least")	`Age >= 18` is FALSE

"**Over** 18" is `Age > 18`. "18 **or over**" and "**at least** 18" are `Age >= 18`.

Joining comparisons: AND, OR, NOT

operator	the statement is TRUE when ...	example
`AND`	**both** parts are TRUE	`Age >= 13 AND Age <= 19`
`OR`	**at least one** part is TRUE	`Day = "Sat" OR Day = "Sun"`
`NOT`	the part is FALSE (**reverses** it)	`NOT Paid`

Every comparison needs **both sides**. Write `Temp >= 18 AND Temp <= 24`. Never write `Temp >= 18 AND <= 24`, and never write `18 <= Temp <= 24`.

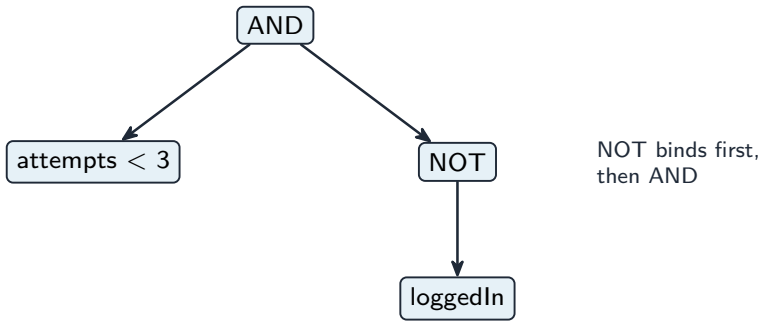
Inside a range, outside a range

- **Inside** a range: the value must pass **both** tests, so use **AND**: `Mark >= 0 AND Mark <= 100`.
- **Outside** a range: the value fails at **one** end, so use **OR**: `Mark < 0 OR Mark > 100`.

Test it with one value. Is 150 outside? `150 < 0` is FALSE, `150 > 100` is TRUE, and FALSE OR TRUE is **TRUE**: correct. With AND it is FALSE for every number: no number is below 0 **and** above 100.

Working out a logic statement

Work in this order: **brackets**, then `NOT`, then `AND`, then `OR`.



Worked example: `Age >= 18 AND (HasTicket OR IsStaff)` *with* `Age` *= 20,* `HasTicket` *= FALSE,* `IsStaff` *= TRUE.*

1. Replace each part with its value: `TRUE AND (FALSE OR TRUE)`.
2. Brackets first: `FALSE OR TRUE` is TRUE, so the statement is `TRUE AND TRUE`.
3. The result is **TRUE**.

A decision in an algorithm

When an algorithm must choose, the logic statement becomes the **condition** of an `IF`.

Worked example: delivery is free for an order of more than 50.00, or for a member.

Otherwise it costs 4.99.

In structured English:

1. If the order total is more than 50.00, or the customer is a member, set the delivery charge to 0.
2. Otherwise set the delivery charge to 4.99.

In pseudocode:

```
IF OrderTotal > 50.00 OR Member = TRUE THEN
    Delivery ← 0
ELSE
    Delivery ← 4.99
ENDIF
```

- The statements between `THEN` and `ELSE` run when the condition is TRUE. The statements between `ELSE` and `ENDIF` run when it is FALSE.
- `Member` is a `BOOLEAN`, so `Member = TRUE` can also be written as `Member`.

Past-paper practice: complete sheets 9.2.1, 11.2 and 11.2.1 before attempting past-paper set 11.2.1.

2 Practice

Now use the methods above. The first questions need one idea each.

2.1 Write the comparison operator for each phrase. [4]

(a) is not equal to

(b) is at least

(c) is less than

(d) is at most

2.2 The variable `X` holds the value 7. State whether each logic statement is TRUE or FALSE. [5]

(a) `X > 7`

(b) `X >= 7`

(c) `X <> 7`

(d) `X < 10 AND X > 5`

(e) `X < 5 OR X = 7`

X [7]

2.3 Write a logic statement for each condition.

(a) `Age` is from 13 to 19. [1]

Age [ ] UserName [ ]

(b) `UserName` is not an empty string and has at most 12 characters. The function `LENGTH()` returns the number of characters in a string. [1]

(c) A ticket is free when `Age` is under 5 or is 80 or more. [1]

2.4 Each logic statement contains an error. Write the correct statement. [4]

(a) `Mark >= 0 AND <= 100` should be TRUE for a mark from 0 to 100.

[ ]

(b) `Speed < 30 AND Speed > 120` should be TRUE for a speed that is **outside** the range 30 to 120.

2.5 `Score` is 45, `Paid` is TRUE and `Banned` is FALSE. Work out whether each logic statement is TRUE or FALSE. Show your working. [4]

(a) `Score >= 50 OR Paid`

`Score` [45] `Paid` [TRUE] `Banned` [FALSE]

(b) `Paid AND NOT Banned AND Score > 45`

2.6 A shop gives a discount of 10% when a customer buys more than 5 items. Complete the pseudocode. [3]

```
IF ............................... THEN
    Discount ← ...............................
ELSE
    Discount ← ...............................
ENDIF
```

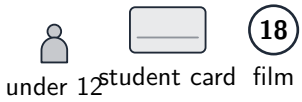
3 Exam-style questions

3.1 A cinema has three rules.

- Rule 1: a customer pays the child price if they are under 12.
- Rule 2: a customer pays the student price if they are from 12 to 17 **and** have a student card.
- Rule 3: a film marked "18" can be watched only by a customer who is 18 or over.

The variables are `Age` (an integer) and `HasCard` (a Boolean).

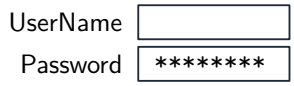
(a) Write a logic statement for **each** rule. [3]



(b) A customer is 17 and has no student card. State which of your three logic statements are TRUE for this customer. [2]

3.2 A new password is **valid** if it has at least 8 characters **and** it is different from the user name. The function `LENGTH()` returns the number of characters in a string.

(a) Write a logic statement that is TRUE when the password is **valid**. Use the variables `Password` and `UserName`. [2]



(b) Write a logic statement that is TRUE when the password is **not valid**. Do **not** use `NOT`. [2]

3.3 An algorithm is described in structured English:

1. Input the total of an order.
2. Input whether the customer is a member (TRUE or FALSE).
3. If the total is more than 50.00, or the customer is a member, the delivery charge is 0. Otherwise the delivery charge is 4.99.
4. Add the delivery charge to the total.
5. Output the total.

Write pseudocode for this algorithm. Declare your variables. [6]

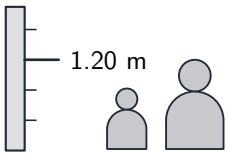
member

3.4 A theme park has a fast ride. A child may go on the ride only if the child is at least 1.20 m tall. A child who is under 10 years old must **also** be with an adult.

(a) Complete the identifier table for this problem. [3]

identifier	data type	description
		the child's height in metres
		the child's age in years
		whether the child is with an adult

(b) Write a logic statement that is TRUE when the child **may** go on the ride. [3]



(c) Write pseudocode that inputs the three values and outputs `"Enjoy the ride"` or `"Sorry, not today"`. Declare your variables. [5]

(d) State what is output for each child. [2]

- Child 1: height 1.35 m, age 8, not with an adult
- Child 2: height 1.25 m, age 11, not with an adult

Solutions

Each answer shows the steps, then the **result**.

2.1

(a) `<>`

(b) "at least" includes the value itself → `>=`

(c) `<`

(d) "at most" includes the value itself → `<=`

2.2

(a) 7 is not greater than 7 → **FALSE**

(b) 7 is equal to 7 → **TRUE**

(c) 7 is equal to 7, so "not equal" → **FALSE**

(d) TRUE AND TRUE → **TRUE**

(e) FALSE OR TRUE → **TRUE**

2.3

(a) inside a range, so AND: `Age >= 13 AND Age <= 19`

(b) `UserName <> "" AND LENGTH(UserName) <= 12`

(c) two separate groups, so OR: `Age < 5 OR Age >= 80`

2.4

(a) the second comparison has only one side

- `Mark >= 0 AND Mark <= 100`

(b) no speed is below 30 **and** above 120 at the same time, so the statement is never TRUE; outside a range needs OR

- `Speed < 30 OR Speed > 120`

2.5

(a) `45 >= 50` is FALSE, and `Paid` is TRUE: FALSE OR TRUE

- **TRUE**

(b) `NOT Banned` is TRUE, but `45 > 45` is FALSE: TRUE AND TRUE AND FALSE

- **FALSE**

2.6

- the condition: `Items > 5` (any sensible identifier)
- the TRUE branch: `Discount ← 0.1` (or 10)
- the FALSE branch: `Discount ← 0`

3.1

(a) rule 1: `Age < 12`

- rule 2: `Age >= 12 AND Age <= 17 AND HasCard` (or `HasCard = TRUE`)
- rule 3: `Age >= 18`

(b) rule 1: `17 < 12` is FALSE; rule 3: `17 >= 18` is FALSE

- rule 2: the age is in the range, but `HasCard` is FALSE, so the AND is FALSE: **none** of the three statements is TRUE

3.2

(a) `LENGTH(Password) >= 8`

- `AND Password <> UserName`

(b) the password is not valid when **either** rule is broken, so the operator becomes OR

- `LENGTH(Password) < 8 OR Password = UserName`

3.3

- the variables declared with suitable types, including a `BOOLEAN` for the member
- the two inputs
- an `IF` with the condition `Total > 50.00 OR Member = TRUE`
- the delivery charge set to 0 in one branch and to 4.99 in the other, with `ELSE` and `ENDIF`
- the delivery charge added to the total **after** the `IF`
- the total is output

```
DECLARE Total : REAL
DECLARE Delivery : REAL
DECLARE Member : BOOLEAN
INPUT Total
INPUT Member
IF Total > 50.00 OR Member = TRUE THEN
    Delivery ← 0
ELSE
    Delivery ← 4.99
ENDIF
Total ← Total + Delivery
OUTPUT Total
```

3.4

(a) one mark for each row with a meaningful name **and** the correct type, for example:

identifier	data type	description
`Height`	`REAL`	the child's height in metres
`Age`	`INTEGER`	the child's age in years
`WithAdult`	`BOOLEAN`	whether the child is with an adult

(b) the height test: `Height >= 1.20`

- the age rule: the child is 10 or over, **or** is with an adult: `(Age >= 10 OR WithAdult)`
- the two parts joined with AND, with the brackets: `Height >= 1.20 AND (Age >= 10 OR WithAdult)`

(c) the three variables declared

- the three inputs
- an `IF` that uses the logic statement from part (b)
- the two outputs, in the correct branches
- `ELSE` and `ENDIF` used correctly

```
DECLARE Height : REAL
DECLARE Age : INTEGER
DECLARE WithAdult : BOOLEAN
INPUT Height
INPUT Age
INPUT WithAdult
IF Height >= 1.20 AND (Age >= 10 OR WithAdult) THEN
    OUTPUT "Enjoy the ride"
ELSE
    OUTPUT "Sorry, not today"
ENDIF
```

(d) child 1: the height test is TRUE, but `8 >= 10` is FALSE and `WithAdult` is FALSE, so the bracket is FALSE → **"Sorry, not today"**

- child 2: the height test is TRUE and `11 >= 10` is TRUE → **"Enjoy the ride"**