Freebooks.pk

Free Notes • Past Papers • Guides for Students

2nd Year Computer Science — Punjab Board

CHAPTER 11

Decision Constructs

Key Concepts · Definitions · Short & Long Questions · MCQs · Diagrams · Exam Tips

This chapter covers Decision Constructs from the 2nd Year (ICS Part-II) Computer Science syllabus of the Punjab Curriculum and Textbook Board (PTB/PCTB). It explains control structures and the selection (decision) statements of C, the simple if, the if-else, the nested if, the if-else if ladder and the switch statement. These notes are prepared by freebooks.pk.

A program often needs to choose a path of execution based on a condition. Decision constructs, also called selection structures, let a program decide which statements to run.

Learning Objectives

- Explain the three control structures: sequence, selection and repetition.
- Define a condition and a compound statement.
- Use the simple if statement.
- Use the if-else statement.
- Use nested if and the if-else if ladder.
- Use the switch statement with case, break and default.

Key Concepts

Control Structures

Control structures are statements that control the flow of execution in a program or function. They let us group individual instructions into a single logical unit that has one entry point and one exit point. Program instructions can be organised into three kinds of control structures: sequence, in which instructions are executed in the order they are written; selection, in which the program chooses between alternatives; and repetition (also called iteration or loop), in which a statement or group of statements is repeated. Every program, simple or complex, is built from these three structures.

Conditions and Compound Statements

A condition is an expression that evaluates to either true, usually represented by 1, or false, represented by 0. Decision constructs use such conditions to choose a path. A compound statement is a group of statements enclosed in opening and closing braces { }, which are treated as a single unit. When more than one statement must be executed as part of a decision, they must be written as a compound statement inside braces; for a single statement the braces are optional.

The Simple if Statement

The if statement is the simplest decision construct. It allows a statement or a set of statements to be executed only when a condition is true. Its general form is if (condition) { statements }. If the condition is true the statements in the if block are executed; if it is false they are skipped and the program continues with the next statement. When the block contains more than one statement the braces are required; for a single statement they are optional.

The if-else Statement

The keyword else is used with if to provide two different choices. Its general form is if (condition) { true block } else { false block }. If the condition is true the if block is executed; if it is false the else block is executed. Exactly one of the two blocks runs. It is important to enclose a compound statement in braces; omitting the braces around a multi-statement if block causes the error Illegal else without matching if.

The Nested if Statement

A nested if statement is an if statement written inside another if statement, and nesting can be done to any level. It is useful when a decision depends on more than one condition, for example to find the largest of three numbers. An advantage of nested if over a sequence of separate ifs is that, once a decision is reached, the remaining conditions are skipped, whereas a sequence of ifs tests every condition. However, deep nesting increases the complexity of the code and can make it hard to read.

The if-else if Statement

When there are more than three alternatives, nested ifs become complex, so the if-else if statement (an if with multiple alternatives) is a better choice. Its conditions are tested one after another until a true condition is found; the statements following that condition are executed and the remaining alternatives are skipped. If a condition is false the next condition is tested, and if all conditions are false the statements after the final else are executed. This ladder structure is clear and easy to read for many alternatives.

The switch Statement

The switch statement selects one of many alternatives by comparing the value of an expression against a list of case labels. The expression and the case labels must be integer or character values, not float or double. The value of the expression is compared to each case; the statements after the first matching case are executed until a break statement is reached, which causes the rest of the switch to be skipped. A default label provides code to run when no case matches. If the break statements are omitted, execution falls through from the matching case to the end of the switch.

Important Definitions

Term	Definition
Control structure	A statement that controls the flow of execution in a program.
Sequence	Executing instructions in the order they are written.
Selection	Choosing which statements to execute based on a condition.
Condition	An expression that evaluates to true (1) or false (0).
Compound statement	A group of statements enclosed in braces { }.
Nested if	An if statement placed inside another if statement.

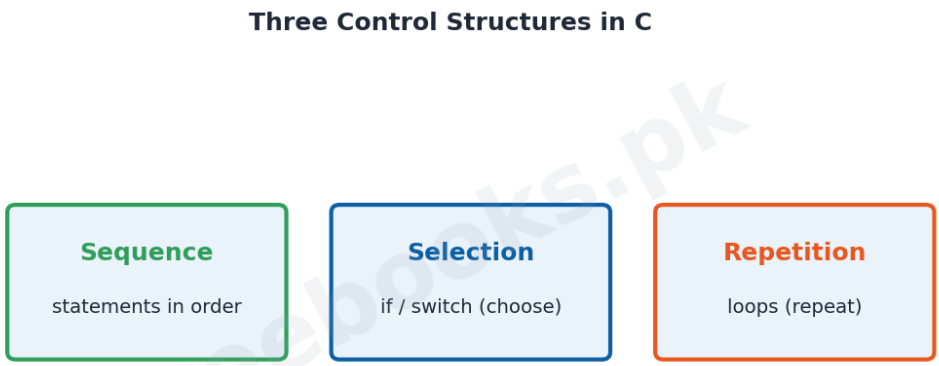
Term	Definition
switch	A selection statement that matches an expression to case labels.
default	The switch label whose code runs when no case matches.

Key Facts & Rules

Item	Detail
Simple if	if (condition) { statements }
if-else	if (condition) { true } else { false }
if-else if	if (c1){} else if (c2){} else {}
switch	switch(expr){ case v1: ... break; default: ... }
Condition value	true = 1, false = 0
case labels	must be integer or character constants
break	ends a case and exits the switch

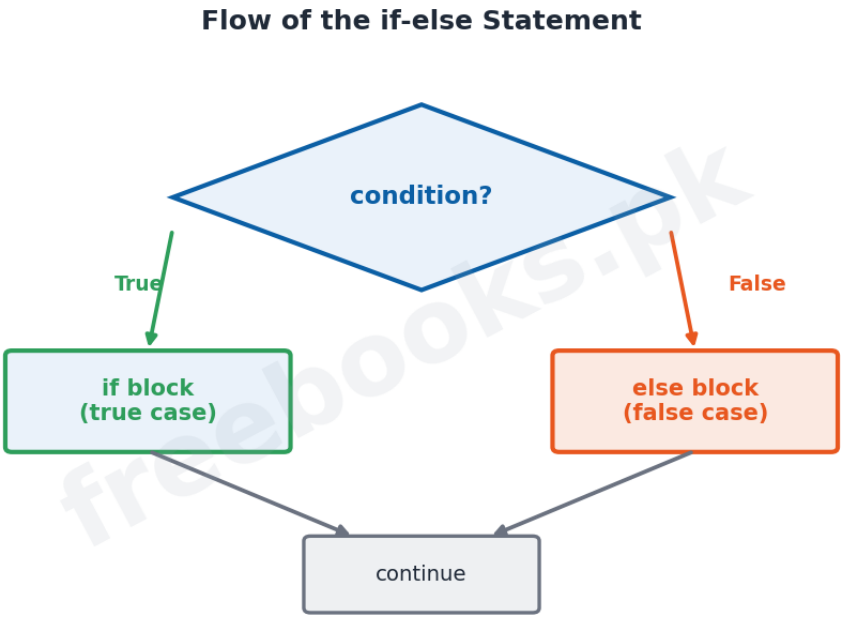
Diagrams & Illustrations

Three control structures: the three control structures used in every program: sequence, selection and repetition.



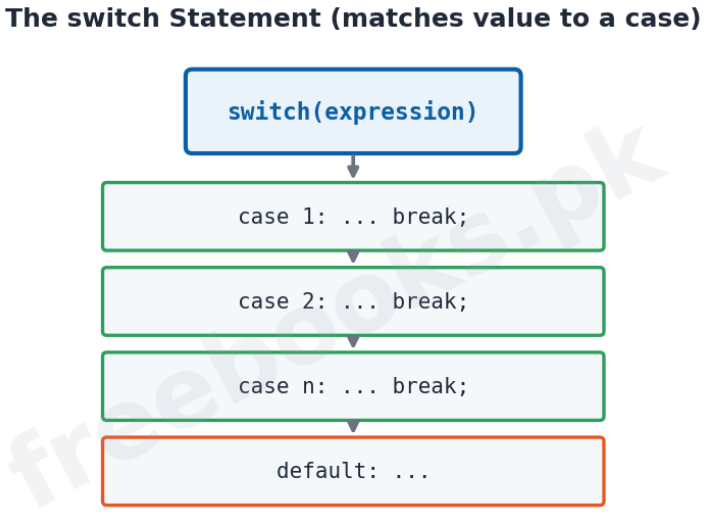
Three control structures

Flow of the if-else statement: how an if-else statement chooses the true block or the false block depending on the condition.



Flow of the if-else statement

The switch statement: how a switch statement matches the expression value to a case and uses break and default.



The switch statement

Solved Examples & Practice

Simple if

if (x > 0) square_root = sqrt(x); calculates the square root only when x is positive.

if-else

if (x >= 0) printf("root exists"); else printf("cannot be calculated"); runs one block or the other.

Nested if for largest

Using if (a > b) then if (a > c) inside it, the program finds the largest of three numbers a, b and c.

switch on a grade

switch(grade){ case 'A': printf("Excellent"); break; default: printf("Try again"); } selects a message by value.

Short Questions & Answers

Q1. Name the three control structures.
Ans. Sequence, selection and repetition (iteration).

Q2. What is a condition?
Ans. An expression that evaluates to true (1) or false (0).

Q3. What is a compound statement?
Ans. A group of statements enclosed in braces { } treated as a single unit.

Q4. What is a nested if statement?
Ans. An if statement written inside another if statement.

Q5. What is the use of the break statement in switch?
Ans. It ends the current case and skips the rest of the switch statement.

Q6. What is the use of the default label in switch?
Ans. It provides code that runs when none of the case labels matches.

Long Questions & Answers

Q1: Explain the simple if, if-else and nested if statements with their general forms.

Decision constructs, also called selection structures, allow a program to choose which statements to execute according to a condition, where a condition is an expression that evaluates to true, represented by 1, or false, represented by 0. The simplest of these is the simple if statement, whose general form is if (condition) followed by a block of statements. The statements in the block are executed only when the condition is true; if the condition is false they are skipped and the program continues with the next statement. When the block contains more than one statement the statements must be enclosed in braces to form a compound statement, although for a single statement the braces may be omitted. The if-else statement extends this idea by using the keyword else to provide a second choice; its

general form is if (condition) { true block } else { false block }. When the condition is true the if block is executed, and when it is false the else block is executed, so that exactly one of the two blocks always runs. It is important to enclose any compound statement in braces, because omitting the braces around a multi-statement if block produces the error message Illegal else without matching if. The nested if statement is an if statement written inside another if statement, and nesting may be carried to any level. It is used when a decision depends on more than one condition, for example when finding the largest of three numbers by testing a against b and then, inside that, a against c. A useful property of the nested if is that once a decision is reached the remaining conditions are skipped, unlike a sequence of separate if statements in which every condition is tested; however, deep nesting makes the code more complex and harder to follow.

Q2: Explain the if-else if statement and compare it with the switch statement.

When a program must choose among more than three alternatives, a nested if statement can become complex and difficult to read, so C provides the if-else if statement, an if statement with multiple alternatives. Its general form is a ladder in which the first condition is tested and, if true, its statements are executed while the rest of the ladder is skipped; if the first condition is false the next else if condition is tested, and so on. If all the conditions are false, the statements following the final else are executed. The conditions are therefore tested in sequence until a true one is found, which makes the structure clear even when there are many alternatives. The switch statement is another way to select one of many alternatives, but it works differently: instead of testing a series of conditions, it compares the value of a single expression against a list of case labels. The expression and the case labels must be of integer or character type; float and double values are not allowed. The value of the expression is compared with each case, and the statements after the first matching case are executed until a break statement is reached, which causes the rest of the switch to be skipped. A default label may be included to provide statements that run when no case matches. If the break statements are omitted, control falls through from the matching case down to the end of the switch, executing all the statements below it. In short, the if-else if ladder is best when the choice depends on ranges or complex conditions, while the switch statement is especially convenient when the selection is based on the exact value of a single integer or character expression.

Q3: Explain the switch statement in detail with its syntax, the role of break and default, and its limitations.

The switch statement is a multi-way selection statement in C that chooses one of many alternatives by comparing the value of an expression against a list of possible values. Its syntax begins with the keyword switch followed by an expression in parentheses, and then a block containing several case labels, each written as case value: followed by one or more statements and usually a break statement, and optionally a default label. When the switch statement runs, the value of the expression is evaluated once and then compared to each case label in turn. The statements following the first case label whose value matches the expression are executed. Execution continues until a break statement is encountered; the

break causes the rest of the switch statement to be skipped so that control passes to the statement after the closing brace of the switch. If the break statements are omitted, then once a matching case is found the program continues executing every statement from that point down to the end of the switch, a behaviour known as fall-through, which is occasionally useful, for example when several case labels such as 'a', 'A' should run the same code. The default label provides statements to be executed when none of the case labels matches the value of the expression; its position is not fixed and it may be written before the first case or after the last case. The switch statement has an important limitation: the case labels must be integral or character constants and the controlling expression must evaluate to an integer or a character, so it cannot be used with float or double values or with ranges of values. Because of this, the switch statement is especially useful when the selection is based on the exact value of a single integer or character expression, while decisions that involve ranges or complex conditions are better handled by an if-else if ladder.

MCQs with Answers

1. The three control structures are sequence, selection and:
(a) compilation (b) repetition (c) declaration (d) execution
Correct Answer: (b) repetition. Repetition.

2. A condition evaluates to:
(a) true or false (b) only true (c) only false (d) a string
Correct Answer: (a) true or false. True or false.

3. A group of statements in braces { } is a:
(a) function (b) compound statement (c) header (d) loop
Correct Answer: (b) compound statement. Compound statement.

4. The simplest decision construct is the:
(a) switch (b) for loop (c) if statement (d) while loop
Correct Answer: (c) if statement. if statement.

5. The keyword used for the false case with if is:
(a) then (b) else (c) break (d) case
Correct Answer: (b) else. else.

6. An if statement inside another if is called:
(a) chained if (b) nested if (c) switch (d) default
Correct Answer: (b) nested if. Nested if.

7. The switch case labels must be:
(a) float or double (b) integer or character constants (c) strings only (d) any value
Correct Answer: (b) integer or character constants. Integer or character constants.

8. The statement that ends a case in switch is:
(a) stop (b) return (c) break (d) exit
Correct Answer: (c) break. break.

9. The label that runs when no case matches is:

(a) case (b) else (c) default (d) final
Correct Answer: (c) default. default.

10. Omitting braces around a multi-statement if block causes:
(a) no error (b) Illegal else without matching if (c) syntax stop (d) infinite loop
Correct Answer: (b) Illegal else without matching if. Illegal else without matching if.

Quick Revision Summary

- Three control structures: sequence, selection, repetition (loops).
- Condition = expression that is true (1) or false (0); compound statement = statements in { }.
- Simple if runs a block only if true; if-else runs one of two blocks.
- Nested if = if inside if; if-else if ladder tests conditions until one is true.
- switch matches an integer/char expression to case labels; break exits, default = no match.
- Use if-else if for ranges/complex tests, switch for exact single values. Notes by freebooks.pk.

Exam Tips

- List and define the three control structures.
- Give the general form of simple if, if-else and nested if.
- Explain the if-else if ladder with how conditions are tested.
- Give the syntax of switch and explain break and default.
- State the limitation on switch case labels (integer/char only).
- Know the error message for a missing brace: Illegal else without matching if.