Start:CST8116 Intro. to Comp. Prog.Week 09 Lesson 01 Selection Structure (Decisions)Week 09 Lesson 01 Selection Structure (Decisions)Selection Structure, also called Decision StructureSingle-alternative selection structureDual-alternative selection structureNested Selection StructureUsing AND and ORTesting Boundary CasesUsing AND and OR togetherNOT operator (negative logic)Nested logic for ranges of valuesSelection Structure: CaseWelcomeThis presentation provides an introduction to the Selection Structure.Selection Structure, also called Decision StructureThe selection structure permits the program to make a decision, based on data and logic, resulting in different sequence structures to be executed.A selection structure starts with a boolean expression, and then takes one of two paths.[1]Most programming languages will have selection structure, either if, if-then-else, and a case structure (called switch in Java).Single-alternative selection structure [1]A simple program can obtain a user input numeric value and determine if it is less than 10 providing output if this is true.start declarations num number output "enter a number" input number if number < 10 then output "number is less than 10" endif output "thank you for using the program"stopDual-alternative selection structure [1]Determine if it is less than 10 or greater than 10:start declarations num number output "enter a number" input number if number < 10 then output "number is less than 10" else output "number is greater than 10" endif output "thank you for using the program"stopWhat about 10?If you missed it, a value that is exactly 10 will be reported by the program as greater than 10. The program only checks for less than 10, otherwise it will report greater than 10, it does not address if the value is exactly 10.Either need to ask the client the program is for, if 10 needs to be in the lower-range, or upper-range i.e. use <= or <or, if the value 10 needs special treatment we may need to nest one if structure within the otherindentation is used to help visualize the structures.Nested Selection Structure (less, greater, equal 10)start declarations num number output "please enter a number" input number if number < 10 then output "number is less than 10" else if number > 10 then output "number is greater than 10" else output "number is equal to 10“ endif endif output "thank you for using the program"stopNested Selection Structure Alternative SolutionsAlternative solutions are possible.Test your learning:Can you re-write the decision structure above to start with checking if the number equals 10 as well as draw a flowchart?Can you re-write the decision structure to start with checking if the number is greater than 10 as well as draw a flowchart?For this trivial example, there will not be (m)any changes to the structure itself, however there are cases where carefully choosing the order of the logic impacts the number of decisions needed to solve the problem.Using AND and ORstart declarations num number output "please enter a number" input number if number >= 1 AND number <= 10 then output "inside range else output "outside range" endifstopBy using the AND, and OR operators we can make more complicated decisions.E.g. is a number is within a range of values, or outside a range of values. E.g. ANDUsing AND and ORstart declarations num number output "please enter a number" input number if number < 1 OR number > 10 then output "outside range" else output "inside range" endifstopBy using the AND, and OR operators we can make more complicated decisions.E.g. is a number is within a range of values, or outside a range of values. E.g. ORTesting Boundary CasesProgrammers cannot test all possible values that could go into a program.Instead, we focus on boundary cases, also called border cases.For a program that verifies that a number is between 1 to 10 this is a sample test plan.Using AND and OR togetherWhen using AND & OR together in expressions remember order-of-operationsFor a movie theater, if the customer is younger than 12 or older than 65, and the movie is rated G a discount is applied to the ticket sale ([1] pp 161)if age <= 12 OR age >= 65 AND rating = “G” then output “Discount applies”endifThe Bug?: Anyone 12 and under will get a discount, true OR false is true.To fix use ( and )if (age <= 12 OR age >= 65) AND rating = “G” then output “Discount applies”endifNOT operator (negative logic)The NOT operator can be used to flip logic which can simplify a selection structure.There are times where you need to take no action if a condition is true, but take action when it is false. For example this program needs to provide a warning if a value is not equal to 42, but it looks strange without NOT: if number = 42 then else output "not 42" endifNo instructions here, looks awkward Compare (NOT) if NOT (number = 42) then output “not 42" endifCompare (Not-equals, <>) if number <> 42 then output "not 42" endifCompare (OR) if number < 42 OR number > 42 then output "not 42" endifNested logic for ranges of valuesWhere an if-statement is evaluated top-down sequentially, and only one branch of logic per if-else is processed, you do not need to check each range of values.Assume that there are letter grades with ranges at a school:Percent Grade Letter Grade Numeric Grade80 – 100 A 4.070 – 79 B 3.060 – 69 C 2.050 – 59 D 1.00 – 49 F 0.0To determine a student grade based on an entered integer number that falls between a range we could use a sequence of statements with AND logic, but should we or can things be simpler (see next slides)Nested logic for ranges of values (AND not needed)if grade >= 80 AND grade <= 100 then output "A"else if grade >= 70 AND grade <= 79 then output "B" else if grade >= 60 AND grade <= 69 then output "C" else if grade >= 50 AND grade <= 59 then ouput "D" else output "F" endif endif endifendifThere is a problem: someone with a score more than 100 would get F.Nested logic for ranges of values (simplified)if grade >= 80 then output "A"else if grade >= 70 then output "B" else if grade >= 60 then output "C" else if grade >= 50 then ouput "D" else output "F" endif endif endifendifIf already know a value is not >= 80, no need to check again, just ask is it >= 70 and so onSelection Structure: CaseThe case Structure (known as switch in Java)This is a specialized selection structure that matches a value against single values.Note: If a decision needs to be made based on a value exactly matching a set of values this is a good choice of decision structureHowever, if you need to make decisions based on ranges of values, the nested if-else is much better.Similar to else, you can provide a default branch of logic.Selection Structure: CaseHere is a simple example where a user enters one of two options in operating a program, based on numeric integer input.start declaration num userOption output "Please enter 1 or 2 for option" input userOption case userOption 1: output "abc" 2: output "xyz" default: output "invalid option selected" endcasestopTypically Magic Numbers, like 1 and 2 above, are replaced with constants e.g. OPTION\_ABC, OPTION\_XYZConclusionIn this lesson we reviewed:Structured Programming OverviewSelection Structure, also called Decision StructureSingle-alternative selection structureDual-alternative selection structureNested Selection StructureUsing AND and ORTesting Boundary CasesUsing AND and OR togetherNOT operator (negative logic)Nested logic for ranges of valuesSelection Structure: CaseReferences[1] Joyce Farrell. 2018. Programming Logic & Design Comprehensive. 9th Ed. Cengage Learning. Chapter 4 pp. 124 to 167[2] Cay Horstmann. 2019. Big Java Early Objects. 7th Ed. Wiley. Chapter 5 pp. 131 to 170