CS2002D PROGRAM DESIGN

Lecture 2

Conditional Control Structures

Objectives

- To understand how decisions are made in a computer
- To understand the relational operators
- To understand the logical operators and, or and not
- To write programs using relational and logical operators
- To write programs that uses two way selection. if… e lse statements
- To write programs that uses multi way selection if…else if ladder and switch case

- What does Control Structures do ?
- What are the different types of execution flow for a program?

Control Structures

- **Control structures** control the flow of execution in a program or function.
- There are three kinds of execution flow:
  - **Sequence**:
    - the execution of the program is sequential. (add/sub/ mul/div of two numbers)
  - **Selection**:
    - A control structure which chooses alternative to execute.
  - **Repetition**:
    - A control structure which repeats a group of statements.
- We will focus on the **selection** control structure.

- **Selection** control structure?
- Example ?

Conditions

- A program may choose among alternative statements by testing the value of key variables.
  - e.g., if **mark is greater than 50**
    print "Pass"
- **Condition** is an expression that is either false (represented by 0)or true (represented by 1).
  - e.g., "**mark is greater than 50**" is a condition.
- Conditions may contain **relational** or **equality operators**, and have the following forms.
  - variable **relational-operator** variable (or constant)
  - variable **equality-operator** variable (or constant)

Operators Used in Conditions

Operator	Meaning	Type
<	Less than	Relational
>	Greater than	Relational
<=	Less than or equal to	Relational
>=	Greater than or equal to	Relational
==	Equal to	Equality
!=	Not equal to	Equality

Examples of Conditions

Operator	**Condition**	**Meaning**
<=	*x* <= 0	*x* less than or equal to 0
<	`Power` < `MAX_POW`	`Power` less than `MAX_POW`
==	`yes_or_no` == 'Y'	`yes_or_no` == 'Y'
!=	`num` != `SETINEL`	`num` not equal to `SETINEL`

- Logical Operators?
- Examples?

Logical Operators

- There are three kinds of **logical operators**.
  - `&&`: and
  - `||`: or
  - `!`: not
- **Logical expression** is an expression which uses one or more logical operators, e.g.,
  - (temperature > 90.0 `&&` humidity > 0.90)
  - `!`(n <= 0 `||` n >= 100).

The Truth Table of Logical Operators

Op 1	Op 2	Op 1 && Op2	Op 1 || Op2
nonzero	nonzero	1	1
nonzero	0	0	1
0	nonzero	0	1
0	0	0	0

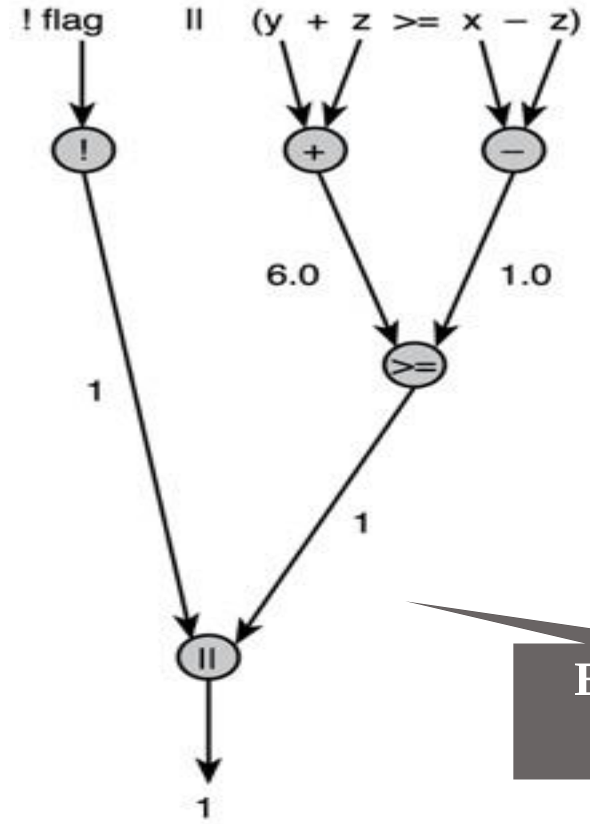
Op 1	! Op 1
nonzero	0
0	1

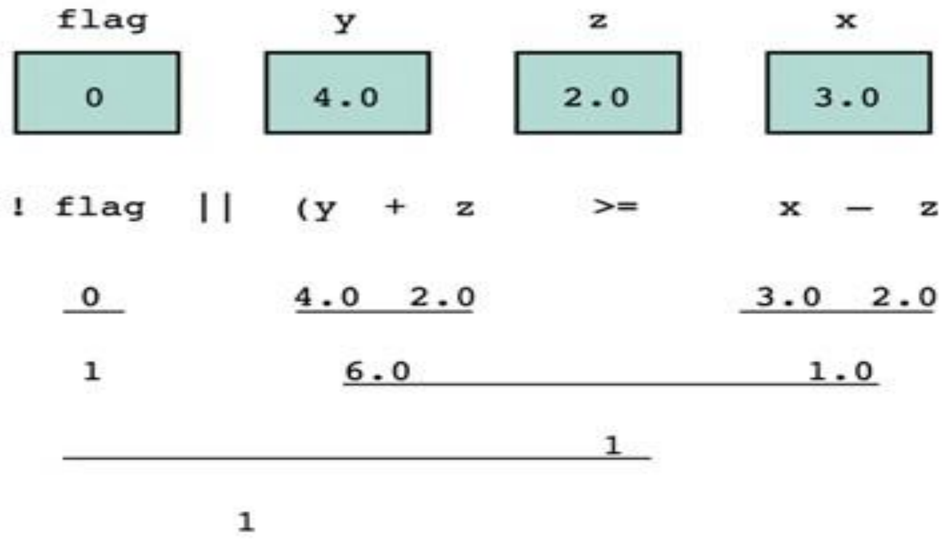
Operator Precedence

- An operator's **precedence** determines its order of evaluation.
- **Unary operator** is an operator that has only one operand.
  - !, +(plus sign), -(minus sign), and &(address of)
  - They are evaluated second only after function calls.

Operator	Precedence
function calls	highest
! + − &	
* / %	
+ −	
< <= >= >	
== !=	
&&	
||	
=	lowest

Evaluation for `!flag || (y + z >= x - z)`





Evaluation tree

The result of this expression is true

DeMorgan's Theorem

- **DeMorgan's theorem** gives us a way of transforming a logical DeMorgan's theorem expression into its complement.
  - The complement of expr$_1$&& expr$_2$ is comp$_1$ || comp$_2$, where comp$_1$ and comp$_2$ are the complement of expr$_1$ and expr$_2$, respectively.
  - The complement of expr$_1$|| expr$_2$ is comp$_1$ && comp$_2$.
- e.g.,age > 25 && (status == 'S' || status =='D')

  is equal to

  (age <=25 ||(status != 'S')&& status!='D')

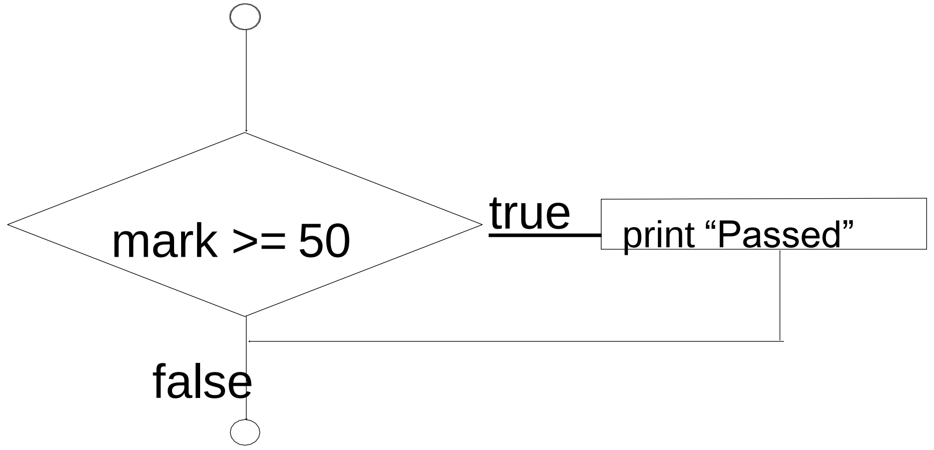
Conditional Statements

- Pseudocode example:

```
If student's mark is greater than or equal to 60
 Print "Passed"
```

- If the condition is **`true`**
  - Print statement executed, program continues to next statement
- If the condition is **`false`**
  - Print statement ignored, program continues

Flowchart of if statement



A decision can be made on any expression.

zero: `false`

nonzero: `true`

Example:

`Mark=60` is `true`

Translation into C

If student's mark is greater than or equal to 50
Print "You have passed in examination"

```
if ( mark >= 50 )
        printf("You have passed in examination!\n");
```

- `if` structure
  - Single-entry/single-exit

Single way selection `if` Statement

- The `if` statement is the primary **selection** control structure. It has a **null else statement.**
- Syntax:

```
if (condition)
      statement;
```

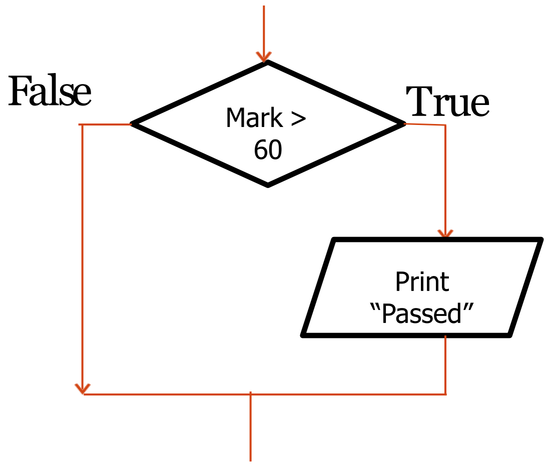
Eg. 1 *If student's mark is greater than or equal to 50*
*Print "*You have passed in examination*"*

```
if(mark > 50)
printf("You have passed in examination!\n");
```

Eg. 2

If quantity is not equal to 0
*calulate price as price * quantity*
print price

```
if (quantity!= 0.0 ){
        price = price * quantity;
        printf("Price=Rs. %8.2f \n",price);
}
```



Two way selection

`if` else Statement

- The `if` else statement is the primary **two way selection control structure.**

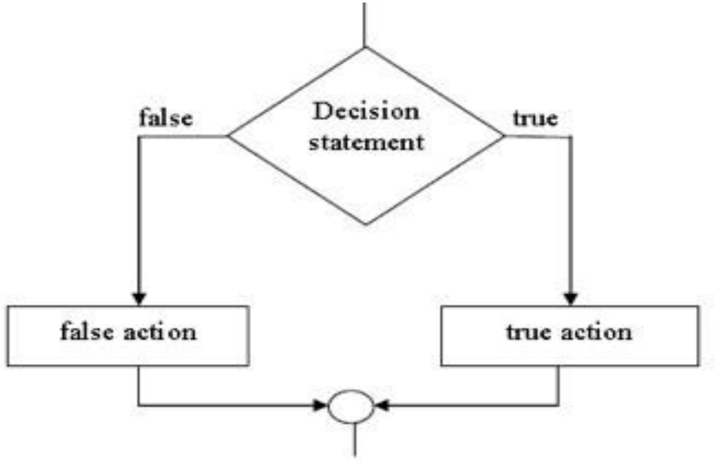
way selection control structure

- Syntax:

```
if(condition)
        statement;
else
        statement;
```

Eg. 1 `if(mark > 50)`

```
                    printf("You have passed in examination! \n");
    else
                    printf("Sorry Try again!\n");
```



Eg. 2

```
if (quantity!= 0.0 ){
                price = price * quantity;
                printf("Price=Rs. %8.2f!\n",price);
               }
else
     printf("Quantity is less than 0.0\n");
```

Single way selection: `if` Statement

- **Write a C program to print the number entered by user only if the number entered is negative.**

```
 void main(){
  int num;
 printf("Enter a number to check.\n");
 scanf("%d",&num);

if(num<0) {
    /* checking whether number is less than 0 or not. */
   printf("Number = %d\n",num);
   }
/*If test condition is true, statement above will be executed, otherwise it will not be executed */
 printf("The if statement in C programming is easy.");
return 0;
}
```

x

Two way selection: `if` else Statement

- **Write a C program to print the number entered by user only if the number entered is positive or negative.**

```
void main(){
 int num;
printf("Enter a number to check.\n");
scanf("%d",&num);
if(num<0) {
     /* checking whether number is less than 0 or not. */
    printf(“Negative Number = %d\n",num);
    }
    else{
/*If test condition is false statement */
 printf(“Positive Number = %d\n",num);
}
return 0;
}
```

Two way selection: `if` else Statement

- **Write a C program to print the number entered by user only if the number entered is even or odd.**

```
 void main(){
  int num;
 printf("Enter a number to check.\n");
 scanf("%d",&num);
if(num%2==0) {
    /* checking whether number is odd or even. */
   printf("Even Number = %d\n",num);
   }
   else{
/*If test condition is false statement */
 printf("OddNumber = %d\n",num);
}
return 0;
}
```

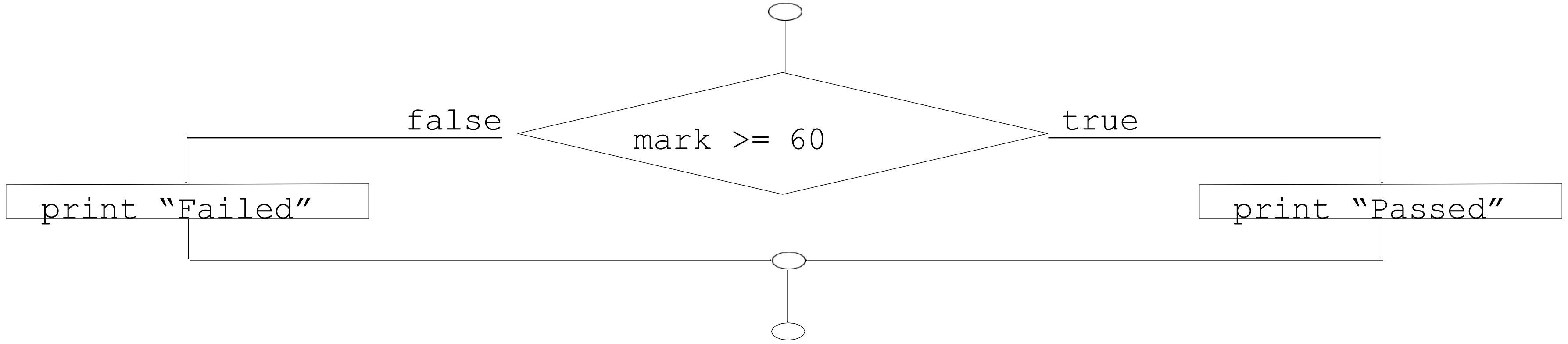
X

`if/else` Selection Structure

- **Ternary conditional operator(?:)**
  - Three arguments (condition, value if true, value if false)
  - Code could be written:

(mark >= 50 ? printf("Passed") :printf("Failed")

Condition Value if true Value if false



Combining condition with logical operators

- If both conditions need to be true use logical AND (&&)operator
  - ➢If first condition is false second condition not evaluate.
- If only one of the condition true use logical OR (||) operator
  - ➢ If first condition is true second condition not evaluated.
- Logical AND (&&)and logical OR (||)act as short circuiting operators.

Combining condition with logical operators

Eg. 1

```
/* if condition 1 is true condition 2 will be evaluated*/  if   (

    (road_status == ‘S’) &&                    (temp>0) )

                printf(“Wet roads ahead!\n”);
 else
                printf(“Drive carefully!\n”);
```

Eg. 2

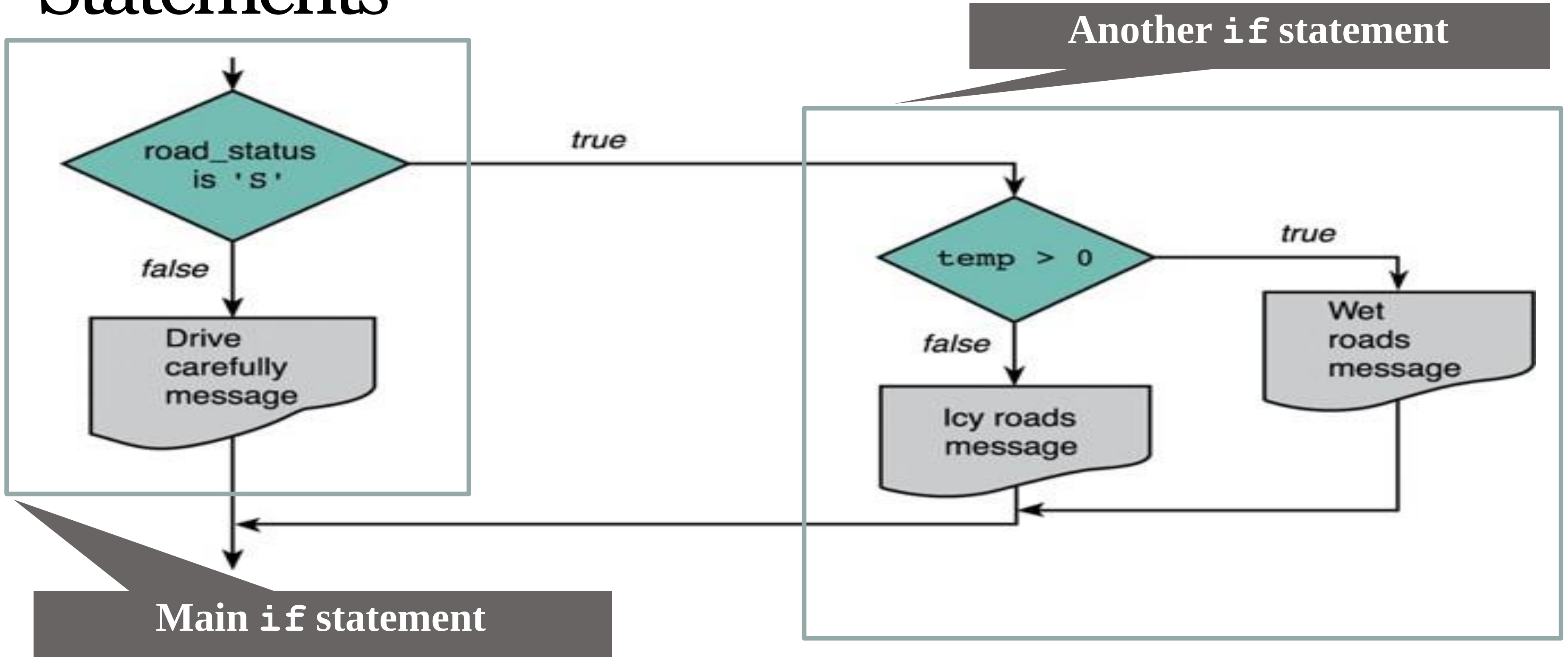
```
/* if condition 1 is true condition 2 will not be evaluated*/
if    (    (road_status == ‘S’) ||                    (temp>0) )

                printf(“Wet roads ahead!\n”);
 else
```

Nested if Statements

- Nested if statement is an ifstatement with **another if statement** as its true task or false task.
- e.g.,

```
if (road_status == ‘S’)  if(temp >  0) {
                              printf(“Wet roads ahead!\n”);
                    }else{
                              printf(“Icy roads ahead!\n”);
                    }
   else
                    printf(“Drive carefully!\n”);
```

An Example for the Flowchart of Nested `if` Statements



Multiway Decision Making statements

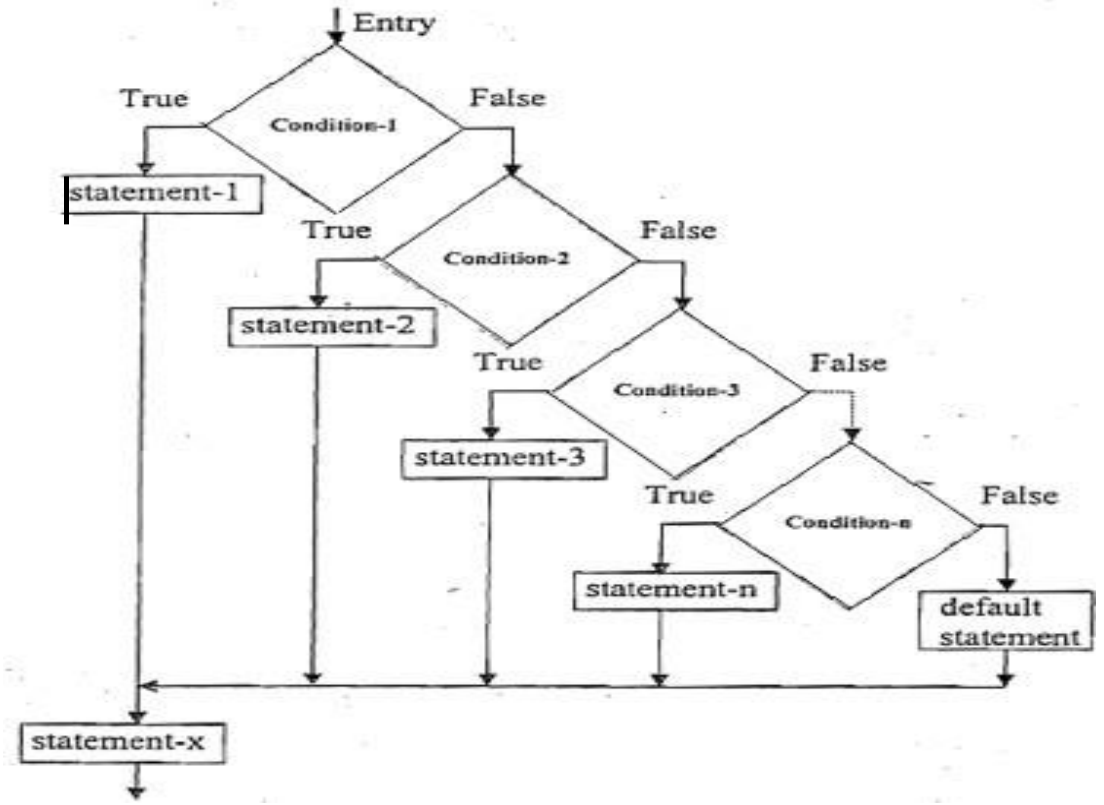
- If there are many alternatives, it is better to use the syntax of **multiway decision**

- **C programming provides two types of multiway control structures.**

  - if else if ladder

  - switch case

Multiway Decision Making statements if else if ladder



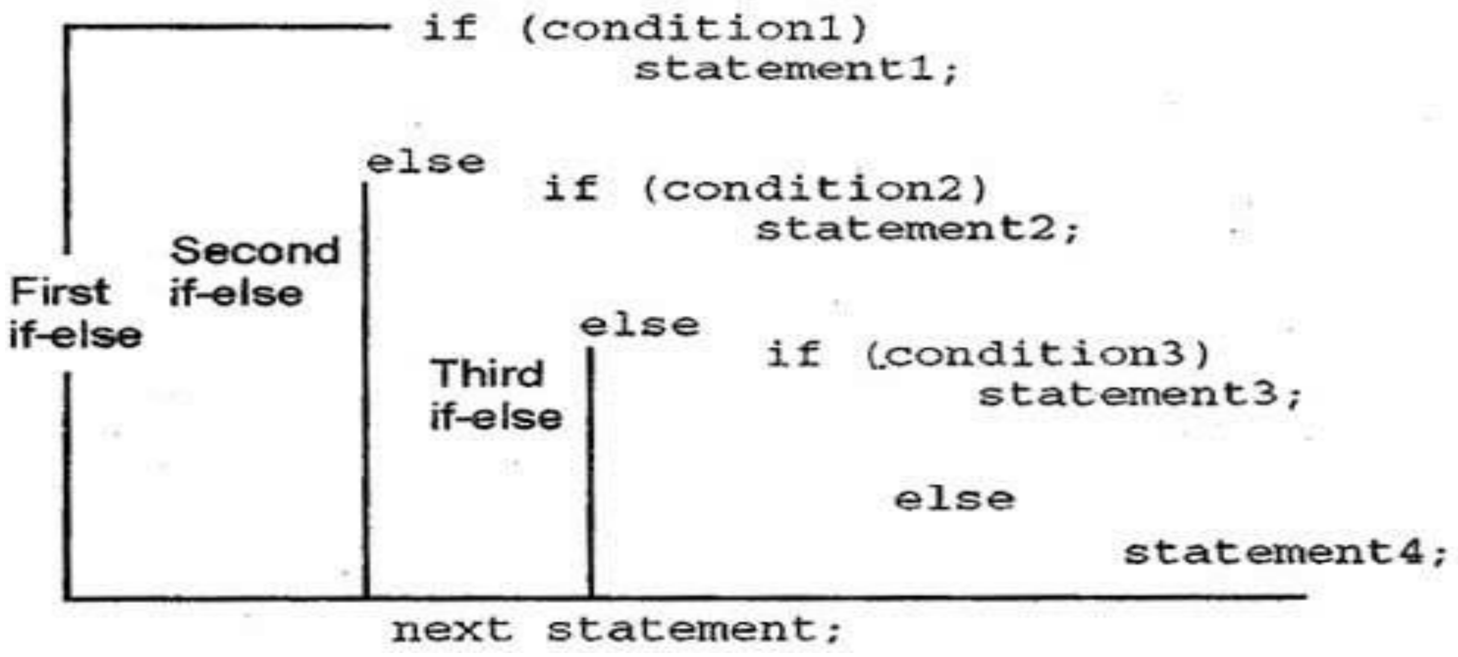
Multiway Decision Making statements if else if ladder

- If there are many alternatives, it is better to use the syntax of **multiway decision.**

- Syntax:

```
if(condition 1) {
           statements; /∗ statements will execute if the condition 1 is true ∗/}
else if(condition 2) {
          statements; /∗ statements will execute if the condition 2 is true ∗/}
else if(condition 3) {
          statements; /∗ statements will execute if the condition 3 is true ∗/}
 else if(condition n) {
          statements; /∗ statements will execute if the condition n is true ∗/}

 else {
          statements; /∗ statements will execute if all conditions are false ∗/
   }
```

Multiway Decision Making statements if else if ladder


```
if (condition1)
        statement1;
else
    if (condition2)
        statement2;
    else
        if (condition3)
            statement3;
        else
            statement4;
next statement;
```

Multiway Decision Making statements if else if ladder

Eg. Write a C program to check if a number is positive negative or zero

```
 void main() {
 int num;
  printf("Enter a number = ");
  scanf("%d",&num);
  if(num>0) {
            printf("Number is Positive");
   }
 else if(a<0) {
            printf("Number is Negative");
     }
else {
             printf("Number is Zero");
}
return 0;
}
```

Multiway-Selection Structure

- **`switch case`**
  - Useful when variable or expression is tested for multiple values
  - Consists of a series of **`case`** labels and an optional **`default`** case
  - **`break`** is (almost always) necessary
    - It is used to terminate a case in the **switch** statement

The `switch` Statement

- The `switch` statement is used to select one of several alternatives when the selection is based on the value of **a single variable** or **an expression**.

```
switch (controlling expression) {
   case label_1:
          statement_1
          break;
   case label_2:
          statement_2
          break;
          ...
   case label_n:
          statement_n
          break;
   default:
          statement_d;
}
```

If the result of this controlling expression matches *label*$_1$, execute staement$_1$ and then break this switch block.

If the result matches none of all labels, execute the default statement$_d$.

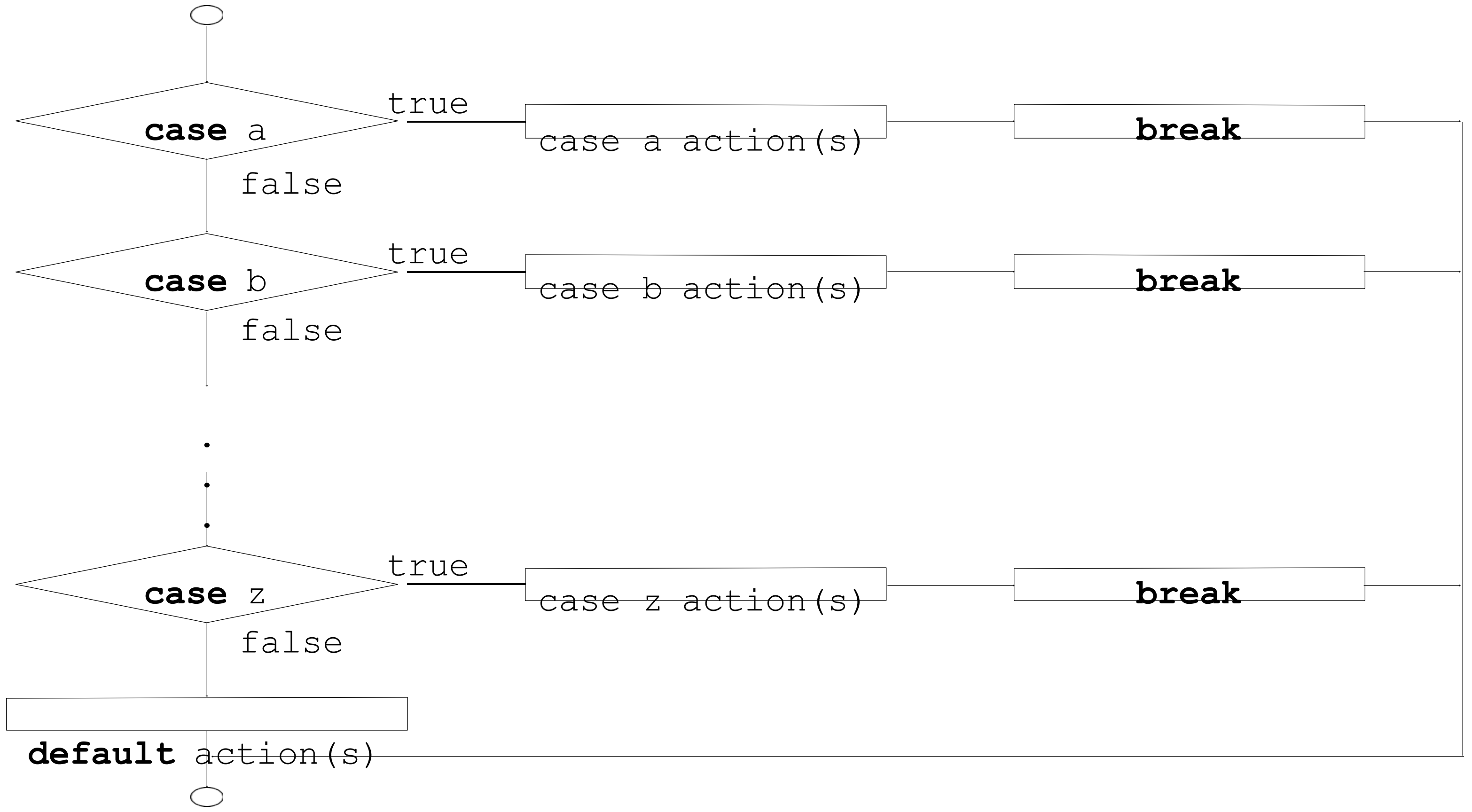
switch case and if else if ladder

```
switch (expression) {
    case val1:        statement
                      break;

    case val2:        statement
                      break;

    case valn:        statement
                      break;

    default:          statement
      ....            break;
}
```

⟷

```
if  (expression == val1)
        statement
else if (expression==val2)
        statement
  ....
else if (expression== valn)
        statement
else
        statement
```

Flowchart switch case



An Example of a `switch` Statement with Type `numeric(int)` Case Labels

Print the day based on the number entered by the user. If any number other than 1-7 is entered say unknown number.

An Example of a `switch` Statement with Type `numeric(int)` Case Labels

```
/* Print the day based on the number entered*/
void main() {
int day;
printf("Enter the day of the week(1-7):");
scanf("%d",&day);
switch(day) {
        case 1: printf("Sunday\n");
                  break;
        case 2: printf("Monday\n");
                   break;
         case 3: printf("Tuesday\n");
                    break;
        case 4: printf("Wednesday\n");  break;
        case 5: printf("Thursday\n");
                 break;
        case 6: printf("Friday\n");
                   break;
        case 7: printf("Saturday\n");
                   break;
        default: printf("Incorrect entry Try again!\n");
    }
```

An Example of a `switch` Statement with Type `char` Case Labels

Write a program to enter the ship name based on the character entered.

If it is B or b	Battleship
If it is C or c	Cruiser
If it is D or d	Destroyer
If it is F or f	Frigate

If it is not any of the letter print unknown ship

Try it yourself

1. Write a program to enter the temperature and print the following message according to the given temperature by using if else ladder statement.

1. T<=0	"Its very very cold".
2. 0< T < 0	"Its cold".
3. 10 < T < =20	"Its cool out".
4. 20< T <=30	"Its warm".
5. T>30	"Its hot".

Try it yourself

2.Write a program that prompts the user to **input the boiling point** in degree Celsius. Using switch case

The program should **output the substance** corresponding to the boiling point listed in the table.

The program should output the message **“substance unknown”** when it does not match any substance.

Substance	Boiling point
Water	100°C
Mercury	357°C
Copper	1187°C
Silver	2193°C
Gold	2660°C

Rules to be followed for switch case

- Case doesn't always need to have order 1, 2, 3 and so on. It can have any integer value after case keyword. Also, case doesn't need to be in an ascending order always, you can specify them in any order as per the need of the program.
- Character labels can be used in switch case.
- **Valid expressions for switch**
  - switch(1+2+23)
  - switch(1*2+3%4)
- **Invalid switch expressions**
    - switch(ab+cd)
    - switch(a+b+c)

    switch(ab+cd) is invalid if ab +cd does not evaluate to either integer or character or enumeration
- Nesting of switch statements are allowed, which means you can have switch statements inside another switch. However nested switch statements should be avoided as it makes program more complex and less readable.

Predict the output

```
void main() {
int i=2;
switch (i) {
     case 1: printf("Case1 ");
     case 2: printf("Case2 ");
     case 3: printf("Case3 ");
     case 4: printf("Case4 ");
    default: printf("Default ");
}
return 0;
}
```

Predict the output

- What will be the output of the following code fragment?

```
int year;
scanf("%d",&year);
if(year%100==0)
{ if(year%400==0)
printf("leap year\n");
}
else
printf("not leap year\n");
```

- if the input given is (i)2000 (ii)1900 (iii) 19

Predict the output

- What will be the output of the following code fragment?

```
int year;
scanf("%d",&year);
if(year%100==0)
{ if(year%400==0)
printf("leap year\n");
}
else
printf("not leap year\n");
```

- **if the input given is (i)2000 (ii)1900 (iii) 19**
- **Ans.**
- **(i)leap year**
- **(ii) No output**
- **(iii) not leap year**

Predict the output

```
void main() {
int i=2;  switch
(i) {
     case 1: printf("Case1 ");  case 2:
    printf("Case2 ");  case 3:
    printf("Case3 ");  case 4:
    printf("Case4 ");  default:
    printf("Default ");
}
return 0;
}
```

Predict the output

```
void main() {
int i=2;  switch
(i) {
     case 1: printf("Case1 ");  case 2:
    printf("Case2 ");  case 3:
    printf("Case3 ");  case 4:
    printf("Case4 ");  default:
    printf("Default ");
}
return 0;
}
```

Output: Case2 Case3 Case4 Default

Reason: No break statement .It will execute the first matching case and then all the case statements below it.

Try it Yourself

- Convert the following if –else loop into switch…case.

```
if ( grade == 'A' || grade == 'a' )
              ++aCount;
          else if ( grade == 'B' || grade == 'b' )
             ++bCount;
         else if ( grade == 'C' || grade == 'c' )
              ++cCount;
        else if ( grade == 'D' || grade == 'd' )
             ++dCount;
      else {
             printf( "Incorrect letter grade entered." );  printf( "
             Enter a new grade.\n" );
```

Summary

- In this lecture we have seen how to alter the flow of a program based on condition.
- The decisions may be two way or multi way.
- The C constructs used for
- Single way : if
- Two way : if else
- Multiway : Nested if else
  - if else if ladder
  - switch case

THANK YOU