Branching: Making Decisions

1. Objective

- Control Statements
- Conditions
- The if Statement
- The if Statement with Compound Statements
- Nested if with multiple-alternative decisions
- The Switch Statement
- Practice: In-class exercises
- Common programming errors

2. Introduction

- The condition which must hold may be logical or relational expression or a Boolean variable.
- The value of the condition must be:
  - **True**
  - **False**

3. Boolean in C

- In the C programming language there is no explicit Boolean type.
- Instead,
  - false is 0 and
  - true is any non-zero value.
- Many C programs use a #define macro to set a Boolean variable to true or false:

```
#define FALSE  0
#define TRUE   1
```

4. Boolean expressions and Conditions

- Used whenever we need conditions such as in the IF statements and WHILE statements

- What is a Boolean expression?
  - It is an expression that has exactly two values: TRUE and FALSE.
  - TRUE and FALSE are called Boolean values
  - The operators used in boolean expression are:
    - **Relational operators**
    - **Logical operators.**

- **Relational operators**:
  - There are six relational operators that compare values of other types and produce a boolean result:

Relational operators	Description
a < b	a is less-than b
a > b	a is greater-than b
a <= b	a is less-than-or-equal-to b
a >= b	a is greater-than-or-equal-to b
a == b	a is equal to b
a != b	a is not equal to b

  - Example:

```
//gcc 5.4.0

#include  <stdio.h>

int main(void)
{
   int A = 10;
   int B = 5;
```

```
    printf("A == B: %d\n", A == B);
    printf("A != B: %d\n", A != B);
    printf("A < B: %d\n", A < B);
    printf("A <= B: %d\n", A <= B);
    printf("A > B: %d\n", A > B);
    printf("A >= B: %d\n", A > B);

    return 0;
}
```

- **Logical operators**:
  - There are also three logical operators:

Logical operator	Description
a && b	Logical AND: true when both of its operands are true
a || b	Logical OR: true when at least one of its two operands are true
!a	Logical NOT (opposite): true when its single operand is false (and false when operand is true)

- Logical Operators Evaluation:
  - AND Operator:

a	b	A && b
0	0	0
0	1	0
1	0	0
1	1	1

  - OR Operator:

a	b	A || b
0	0	0
0	1	1
1	0	1
1	1	1

  - NOT Operator:

a	!a
0	1
1	0

- Example:

```
//gcc 5.4.0
#include  <stdio.h>
int main(void)
{
   int A = 10;
   int B = 5;
   int C = 100;
   int D = 50;
   //Rational Operators
   printf("\nRational Operators Examples\n");
   printf("A == B: %d\n", A == B);
   printf("A != B: %d\n", A != B);
   printf("A < B: %d\n", A < B);
   printf("A <= B: %d\n", A <= B);
   printf("A > B: %d\n", A > B);
   printf("A >= B: %d\n", A > B);
   //Logical Operators
   printf("\nLogical Operators Examples\n");
   printf("(A == B) && (C <= D): %d\n", (A == B) && (C <= D));
   printf("(A != B) || (C >= D): %d\n",  (A != B) || (C >= D));
   printf("!(A == B): %d\n",  !(A == B));
   printf("!(A == B): %d\n",  !(A == B));
   printf("B: %d\n", B);
   printf("C: %d\n", C);
   A = B || C;
   printf("B || C: %d\n", A);
   D = A && B;
   printf("A && B: %d\n", D);

   return 0;
}
```

5. Short-Circuit Evaluation

- "Short circuit" tests save time
- It evaluates the right operand only if it needs to do so. In the following example:

```
m != 0 &&  ((a > b) || (c <= 20)) && (a+b > 0)
```

If n is 0, the right hand operand of && is not evaluated at all because n != 0 is false.

Remember that

false && anything

is always false and the rest of the expression no longer matters.

- C uses short-circuit evaluation for the usual Boolean operators (&& and ||),
- It also provides bitwise Boolean operators that are not short circuit (& and |)

- Example:

```
//gcc 5.4.0
#include  <stdio.h>
// Short-Circuit Evaluation
int main(void)
{
   int A = 5;
   int B = 12;
   int C = 6;
   int D = 50;

   printf("B: %d\n", B);
   printf("C: %d\n", C);
   D = A || B;
   printf("A || B: %d\n", D);
   D = A | B;
   printf("A | B: %d\n", D);
   D = A && B;
   printf("A && B: %d\n", D);
   D = A & B;
   printf("A & B: %d\n", D);

   return 0;
}
```

6. Precedence Rules

Convention	Description	Explanation
( )	Items within parentheses are evaluated first.	In !(age > 16), age > 16 is evaluated first, then the logical NOT.
!	Next to be evaluated is **!**.	
*** / % + -**	Arithmetic operator are then evaluated using the precedence rules for those operators.	z - 45 < 53 is evaluated as (z - 45) < 53.
< <= > >=	Then, relational operators < <= > >= are evaluated.	x < 2 || x >= 10 is evaluated as (x < 2) || (x >= 10) because < and >= have precedence over ||.
== !=	Then, the equality and inequality operators == *!*= are evaluated.	x == 0 && x >= 10 is evaluated as (x == 0) && (x >= 10) because < and >= have precedence over &&.
&	Then, the bitwise AND operator is evaluated.	x == 5 | y == 10 & z != 10 is evaluated as (x == 5) | ((y == 10) & (z != 10)) because & has precedence over |.
|	Then, the bitwise OR operator is evaluated.	x == 5 | y == 10 && z != 10 is evaluated as ((x == 5) | (y == 10)) && (z != 10)) because | has precedence over &&.
&&	Then, the logical AND operator is evaluated.	x == 5 || y == 10 && z != 10 is evaluated as (x == 5) || ((y == 10) && (z != 10)) because && has precedence over ||.
||	Finally, the logical OR operator is evaluated.	

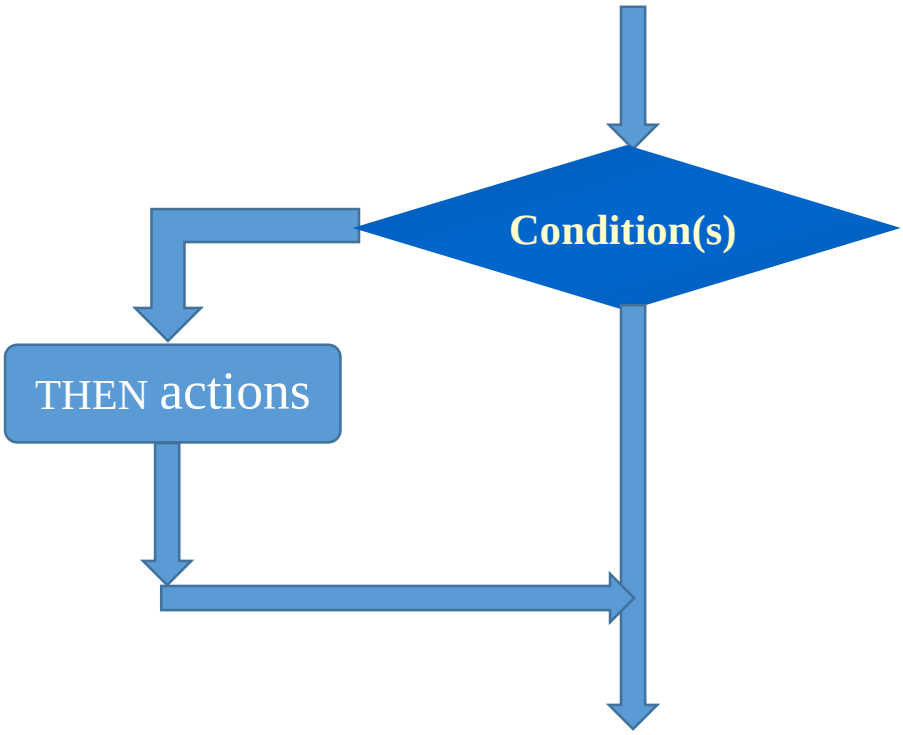
7. Questions/Practice

1) numPeople >= 10
    - true
    - false
2) (numPeople >= 10) && (numCars > 2)
    - true
    - false
3) (numPeople >= 20) || (numCars > 1)
    - true
    - false
4) !(numCars < 5)
    - true
    - false
5) !(userKey == 'a')
    - true
    - false
6) userKey != 'a'
    - true
    - false
7) !((numPeople > 10) && (numCars > 2))
    - true
    - false
8) (userKey == 'x') || ((numPeople > 5) && (numCars > 1))
    - true
    - false

8. One-Way Selection: if Statement

- One alternative: “Things to do” if a condition is TRUE

- Flowchart:



- C implementation: **if statement**

```
if (expression) {
   // Statements to execute when the expression is true (first branch)
}
```

- Example:

```
//gcc 5.4.0

#include  <stdio.h>


int main(void)
{
   double grade;
   int groupSize;
   int numPeople;
   printf("Enter your grade:\n");
   scanf("%lf\n", &grade);
   if (grade > 95)
      printf("A Student:%c\n", 'A');

   return 0;
}
```

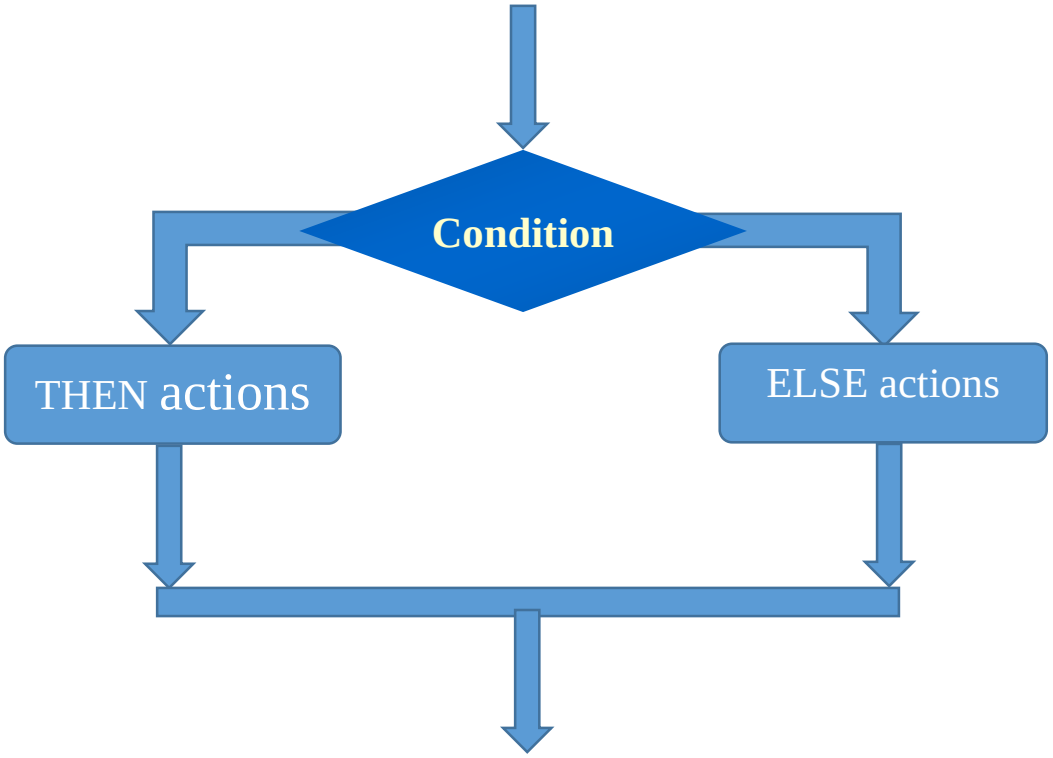
- Another Example: Curly Brackets after the condition.

```
//gcc 5.4.0

#include  <stdio.h>

// Test the following program with salary > 100000 and salary < 100000
int main(void)
{
   double salary = 95000;
   double  bonus = 10000;
   if (salary > 100000)
      salary += bonus;
      printf("Congratulations on your salary increase\n");

   printf("Salary is: %lf\n", salary);
}
```

- Questions/Practice
  - Write a C program that computes student discounts. Apply a 15% discount if the age of a person is less than 25. You program should have the following variable:
    - age
    - price

9. Two-Way Selection: if-else statement

- Two alternatives: “Things to do” if the condition is true, a different “things to do” if the condition is false
- Flowchart:



- C implementation: **if-else statement**

```
if (expression) {
  // Statements to execute when the expression is true (first branch)
}
else {
  // Statements to execute when the expression is false (second branch)
}
```

- Example:
  - What happens if you would like to print a grade different that A in the program we discussed in the previous section?

```
//gcc 5.4.0

#include  <stdio.h>


int main(void)
{
   double grade;
   int groupSize;
   int numPeople;
   printf("Enter your grade:\n");
   scanf("%lf\n", &grade);
   if (grade > 95)
      printf("A Student:%c\n", 'A');
   else
      printf("Not A student\n");
   return 0;
}
```

- Example: Write a C program that computes the following:

  If numPeople is greater than 10, execute groupSize = 2 * groupSize. Otherwise, execute groupSize = 3 * groupSize and also numPeople = numPeople - 1.

```
//gcc 5.4.0

#include  <stdio.h>

int main(void)
{
   int groupSize;
   int numPeople;
   scanf("%d\n", &groupSize);
   scanf("%d\n", &numPeople);
   printf("Group Size Before:%d\n", groupSize);
   printf("Number of People Before:%d\n", numPeople);

   if (groupSize >= 10){
      groupSize *=2;
   } else {
      groupSize *= 3;
      --numPeople;
   }
   printf("Group Size After:%d\n", groupSize);
   printf("Number of People After:%d\n", numPeople);
   return 0;
}
```

10. Multiple Selections

- A series of different “things to do” if a series of conditions hold
- C implementation: if-else if-else structure
- Alternate C implementation: Nested if statements

```
if (expr1) {

}
else if (expr2) {

}

...

else if (exprN) {

}
else {

}
```

- Example:

```
//gcc 5.4.0

#include  <stdio.h>


int main(void)
{
   double grade;
   int groupSize;
   int numPeople;
   printf("Enter your grade:\n");
   scanf("%lf\n", &grade);
   if (grade >= 90)
      printf("Your grade is:%c\n", 'A');
   else if (grade >= 80)
      printf("Your grade is:%c\n", 'B');
   else if (grade >= 70)
      printf("Your grade is:%c\n", 'C');
   else if (grade >= 60)
      printf("Your grade is:%c\n", 'D');
   else
      printf("Your grade is:%c\n", 'F');
   return 0;
}
```

11. Ternary operator

- It is also called the conditional expression

- It is used to reduce the number of line of codes and speed up your implementation.

- C Implementation:

  <condition> ? <expression1> : <expression2>

  - The **expression** on the left is evaluated first:
    - If it evaluates to "**true**", **expression1** is evaluated and its result is returned.
    - If it evaluates to "**false**", **expression2** is evaluated and its result is returned.

- Example:

```
//gcc 5.4.0
#include  <stdio.h>
int main(void)
{
   int a , b;
   a = 10;
   printf( "Value of a is %s\n", (a % 2) ? "Odd": "Even" );
   b = 11;
   printf( "Value of b is %s\n", (b % 2) ? "Odd": "Even" );
   return 0;
}
```

- Questions/Practice:
  - Convert the following code to a condition expression:

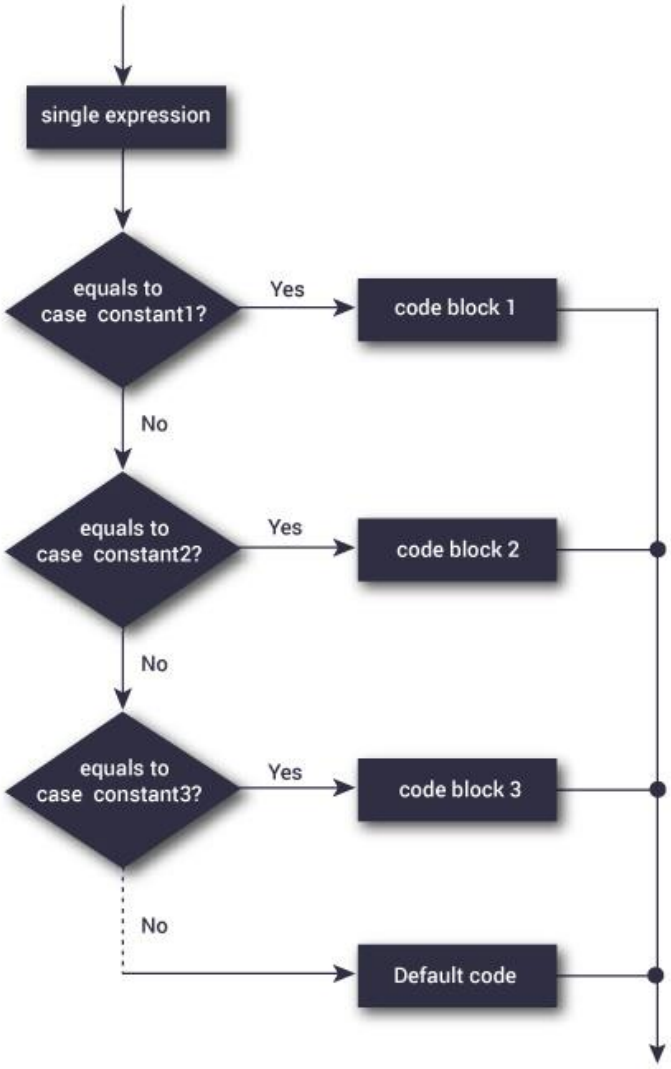
```
if (x > 50) {
   y = 50;
}
else {
   y = x;
}
```

  - Convert the following code to a condition expression:

```
if (x == 0 ||  x ==1) {
   print(“Binary\n”);
}
else {
  printf(“No Binary\n” ;
}
```

12. Switch Structures

- A series of different “things to do” depending on the value of a particular
- mostly used when we have number of options (or choices) and we may need to perform a different task for each choice.
- Flowchart: (Figure from www.programiz.com)



- C Implementation:

```
switch (variable or an integer expression)
{
   case constant1:
      // code to be executed if n is equal to constant1;
      break;

   case constant2:
      // code to be executed if n is equal to constant2;
      break;
      .
      .
      .
   default:
      // code to be executed if n doesn't match any
constant
}
```

- Example:

```
//gcc 5.4.0
#include  <stdio.h>
int main(void)
{
   int myVal;
   printf("0: Main Menu\n");
   printf("0: Open File\n");
   printf("0: Print File\n");
   printf("0: Quit\n");
```

```
    printf("\n\nChoose an option: \n");
    scanf("%d\n", &myVal);
    switch (myVal) {
      case 0:
        printf("Main Menu\n");
        break;
      case 1:
        printf("Open File\n");
        break;
      case 2:
        printf("Print File\n");
        break;
      case 3:
        printf("Quit\n");
        break;
      default:
        printf("No Option\n");
     break;
    }
    return 0;
}
```

13. Character Operations

- C provides several built-in functions to working with characters
- These functions can be found in the ctype.h library:

#include <ctype.h>

isalpha(c)	true if alphabetic: a-z or A-Z	isalpha('x') // true isalpha('6') // false isalpha('!') // false

isdigit(c)	true if digit: 0-9.	isdigit('x') // false
		isdigit('6') // true

isspace(c)	true if whitespace.	isspace(' ') // true
		isspace('\n') // true
		isspace('x') // false

toupper(c)	Uppercase version	toupper('a') // A
tolower(c)	Lowercase version	toupper('A') // A
isalnum(c);	The function returns nonzero if c is alphanumeric	
iscntrl(c);	The function returns nonzero	

	if c is a control chracter	
ispunct(c);	The function returns nonzero if c is punctuation	
isupper(c);	The function returns nonzero if c is upper case character	
Islower(c)	The function returns nonzero if c is lower case character	

- Example: Check your Character

//gcc 5.4.0

```
#include  <stdio.h>

int main(void)
{
   char myChar;
   printf("Please enter a single Character:\n");
   scanf("%c\n", &myChar);
   printf("The entered charcter is:%c\n",  myChar);
   //Check if the character is a digit
   if (isspace(myChar))
```

```
        printf("You have entered a space character\n");
    else if (ispunct(myChar))
        printf("You have entered an punctuation character");
    else if (iscntrl(myChar))
        printf("You have entered a control chacacter");
    else if (isalnum(myChar)){
        printf("You have entered an alphanumeric character AND ");
        if (isdigit(myChar))
            printf("your character is a digit\n");
        else if(isalpha(myChar))
                if (islower(myChar))
                    printf("your character is an lower case alphabetic
character\n");
                else
                    printf("your character is an upper case alphabetic
character\n");

    }
    return 0;
}
```

- Programming Assignment: Convert the above program using **Switch statement.**

14. Questions/Practice

- Given the following code:

```
//gcc 5.4.0
#include <stdio.h>
main(void)
{
int myChar = 0;

printf("enter a value:");
scanf("%d", &myChar);

if (myChar == 1)
if (myChar == 0)
printf("inside if\n");
else
printf("inside else if\n");
else
printf("inside else\n");
}
```

  - What is the output of the following code if the user enter the following values:

User Input	Output of your program
0	
1	
2	
a	

- Given the following code:

```
#include <stdio.h>
void  main()
{
int myVar;
printf("enter a value:\n");
scanf("%d", &myVar);
if (myVar--)
printf("True--\n\n");
if (myVar++)
printf("False--\n\n");
}
```

  - What is the output of the following code if the user enter the following values:

User Input	Output of your program
0	
1	
-1	
a	

- Given the following code:

```
#include <stdio.h>
void  main()
{
int myVar;
printf("enter a value:\n");
scanf("%d", &myVar);
if (--myVar)
printf("True--\n\n");
if (++myVar)
printf("False--\n\n");
}
```

  - What is the output of the following code if the user enter the following values:

User Input	Output of your program
0	
1	
-1	
a	

- Given the following code:

```
#include <stdio.h>
int main()
{
   int x = 2, y = 0;
   scanf("%d\n", &y);
   int z = (++y) ? y == 1 && x : 0;
   printf("x=%d, y=%d, z=%d\n", x,y,z);
   return 0;
}
```

  - What is the output of the following code if the user enter the following values:

User Input	Output of your program
0	
1	
2	
-1	

- Given the following code:

```
 //gcc 5.4.0
#include <stdio.h>
#define  one   '1'
#define   two   '2'
void main()
{
   char myChar;
   printf("Enter a value between 1 to 2:");
   scanf("%c", &myChar);
   switch (myChar)
   {
      case one:
         printf("%c\n",one);
         break;
      case two:
         printf("%c\n",two);
         break;
   }
}
```

  - What is the output of the following code if the user enter the following values:

User Input	Output of your program
0	
1	
2	
a	

- Given the following code:

```
//gcc 5.4.0
#include <stdio.h>

void main(void)
   {
      int myChar;
      printf("enter a value btw 1 to 2:");
      scanf("%d", &myChar);
      switch (myChar)
      {
      case 1:
         printf("1\n");
         break;
         printf("Hi");
      default:
         printf("2\n");
      }

}
```

  - What is the output of the following code if the user enter the following values:

User Input	Output of your program
0	
1	
2	
a	

- Write a C program enters the cost price and selling price of a product and computes any profit or loss. If the cost price is greater than selling price then there is a loss otherwise profit. Give, a Selling Price (SP) and a Cost Price (CP), the formula to calculate profit or loss:

  Profit = S.P - C.P
  Loss = C.P - S.P

- Write a C program to check whether a triangle is equilateral, scalene or isosceles triangle:
  - A triangle is Equilateral Triangle if all its sides are equal.
  - A triangle is Isosceles Triangle if its two sides are equal.
  - A triangle is Scalene Triangle if none of its sides are equal.

- Simple Calculator: Write a C program that implements a simple calculator to perform addition, subtraction, multiplication or division depending the input from user.

- Write a c program to enter any alphabet and check whether alphabet is vowel or consonant using switch case. C program for checking vowels and consonants using switch.