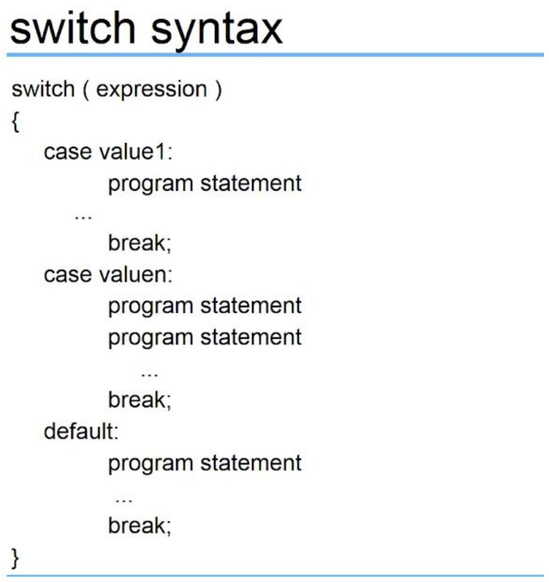
Switch Statements:

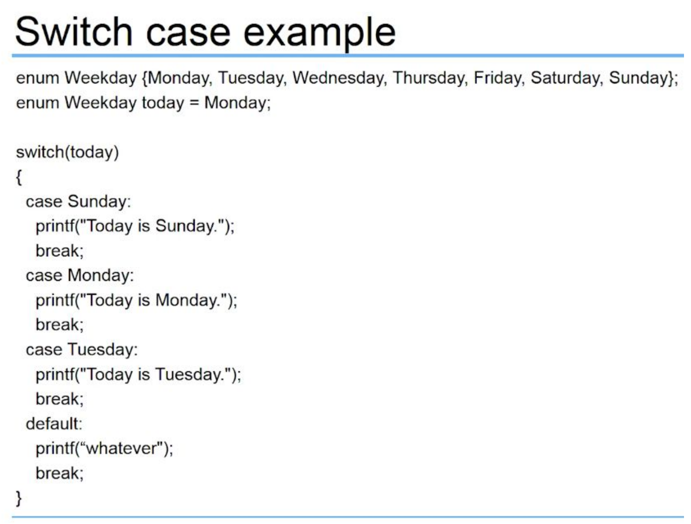
- The conditional operator and the if else statements make it easy to write programs that choose between two alternatives.
- However , many times a programmer needs to choose one of several alternatives.
    - ➔ You can do this by using if else if … else
    - ➔ Tedious , prone to errors.
- When the values of a variable is successively compared against different values use the switch statement.
    - ➔ More convenient and efficient.



switch syntax

```
switch ( expression )
{
    case value1:
            program statement
        ...
            break;
    case valuen:
            program statement
            program statement
                ...
            break;
    default:
            program statement
             ...
            break;
}
```

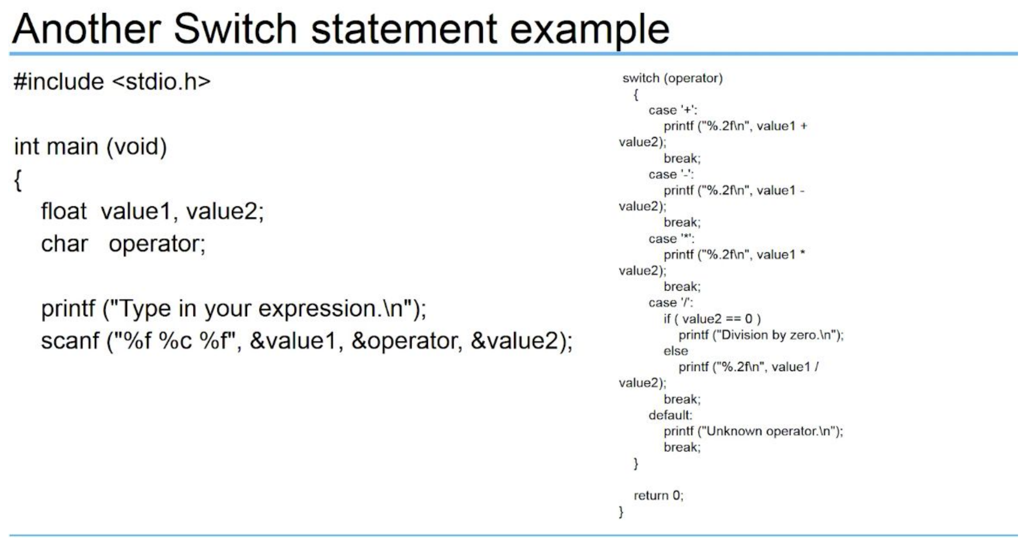
- The expression enclosed within the parentheses is successively compared against the values: value1, value2, … valuen
    - ➔ Cases must be simple constants or constant expressions.
- If a case is found whose value is equal to the value of expression than the statements that follow the case are executed.
    - ➔ When more than one statement is included, they do not have to be enclosed within braces.
- The break statement signals the end of a particular case and causes execution of the switch statement to be terminated.
    - ➔ Include the break statement at the end of every case.
    - ➔ Forgetting to do so for a particular case causes program execution to continue into the next case.
- The special optional case called default is executed if the value of expression does not match any of the case values.
    - ➔ Same as a “fall though” else



Switch case example

```
enum Weekday {Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday};
enum Weekday today = Monday;

switch(today)
{
  case Sunday:
    printf("Today is Sunday.");
    break;
  case Monday:
    printf("Today is Monday.");
    break;
  case Tuesday:
    printf("Today is Tuesday.");
    break;
  default:
    printf(“whatever");
    break;
}
```



Another Switch statement example

```
#include <stdio.h>

int main (void)
{
   float  value1, value2;
   char   operator;

   printf ("Type in your expression.\n");
   scanf ("%f %c %f", &value1, &operator, &value2);
```

```
 switch (operator)
   {
      case '+':
         printf ("%.2f\n", value1 +
value2);
         break;
      case '-':
         printf ("%.2f\n", value1 -
value2);
         break;
      case '*':
         printf ("%.2f\n", value1 *
value2);
         break;
      case '/':
         if ( value2 == 0 )
            printf ("Division by zero.\n");
         else
            printf ("%.2f\n", value1 /
value2);
         break;
      default:
         printf ("Unknown operator.\n");
         break;
   }

   return 0;
}
```

goto statement

- The goto statement is available in C.
  - ➔ Has two part = the goto and a label name.
  - ➔ Label is named following the same convention used in naming a variable.

```
goto part2;
```

- For the above there must be another statement bearing the part2 label.
- You should never need to use the goto statement.
  - ➔ If you have a background in older versions of FORTRAN or BASIC, you might have developed programming habits that depend on using goto .

goto example

- Form:

```
goto label;
   .
   .
   .
label : statement
```

- Example:

```
top : ch = getchar();
     .
     .
     .
if (ch != 'y')
   goto top;
```