07/JULY/2024 WEDNESDAY

classmate Date Page

WB Q20, 22, 23 solved

Lec#13 Flow control statements (4)

switch statements

↳ keyword

- to create selection statement with multiple choices.
- Multiple choices can be provided by using another keyword : case.

```
switch (n)
{
    case 1: // code to be executed if the value of n : 1 followed by a break
    case 2: // if value of n : 2 followed by break.

    default: // code to be executed if the value of n is other than 1 & 2
}           break;
```

default is executed when the value of n doesnot match any case.

```
switch(expression/condith)
    {
        case Constant1 : statements;
                         break;
        case constant2 :
                         break;
        case constant3 :
                         break;
        default        :
                         break;
    }
```

depending upon the expression which matches 1 of the case ⇒ that following case is executed and then break ⇒ comes out of the switch

Q
```
int i=2;
switch(i)
    {
        case 1: printf("one");
                break;
        case2 : pf("Two");
                break;
        default : pf("Default");
                break;
```

1. in first step there is matching of expression with the case
2. when the ~~number is matched~~ now the case ~~statements~~ labels are of no use
3. the code corresponding to the case is executed.

i.e

eg

```
switch(2)
{
case 1 : ____
case 2 : pf("Two");
         break;
default : ____
}
```

significance ends once case is selected.

compiler will now run the code from this point

Q

```
int i = 2;
switch(i)
{
case 2 : pf("Two");
case 1 : pf("One");
default : pf("Hello");
}
```

o/p : Two one Hello

∵ No break means sequential execution of the remaining statements as well.

This is falling through all the cases until Break encountered.

Break is optional.

```
switch(2) {
    case 3 : pf("Three");
    case 2 : pf("Two");
    case 1 : pf("One");
             break;
    default : pf("Hello");
}
```

o/p : Two one

```
switch(expression)
{

}
```

what can be expression
↳ anything which can evaluate to a constant integer.

```
switch (2)
    {
    ____
    ____
    }
```

```
switch(1.2)
    {
    ____
    ____
    }
```

compiler उस के साथ मारेगा

Expression → 1+2*3 = 7

```
int a=10,b=20;
a+b*2
```

```
switch('A')
    {
    }
```

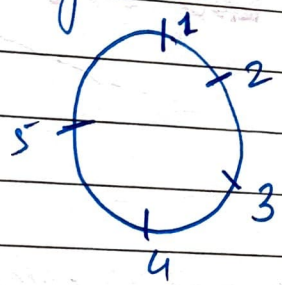
```
switch('A'+2)
    {
    }
```

switch (1.2 + 2)
{
}

Udd ke laath

(2) expression inside switch → 1+2*3
→ a+b * 2
→ 'A'
→ 'A' + 2
→ !2
→ 2&&3
→ 2||3&&4
→ pf ("Vinit")
→ sf ("%d", &x);

Fan → Regulator



Every value of the regulator has different function → i.e different speeds

No → 2 values can have same speeds

ie → if 2 values have same we say regulator is खराब

similarly case labels can't be duplicate.

```
int i = 2;
switch (i)
    {
    case 4 : [ ]
             break;

    case 2+2 : [ ]
               break;

    default : [ ]
              break;
    }
```

compiler ud ke kaatu marega

⇒ not necessary

without the last break also switch is going to end anyways.

★★★ (3) Duplicate case labels are not allowed.

```
int a=2, b=4, c=10;
switch(2+2)
{
    case a : pf("%d", a);
             break;
    case b : pf("%d", b);
             break;
    case c : pf("%d", c);
             break;
    default : pf("Hello");
              break;
}
```

compiler udd ke laath maarega

*** (4) case labels cannot be variables
↳ They can be any constants or expression with all constants (literals)

2, 2+2, 2+3*4, !2, 2||3,
'A', 'A'+2, a+b ×××

(2)
```
int i=2;
switch(i)
{
    case 2:
        break;
    case 4:
        break;
    case 65;
        break;
    case 'A' :
        break;
    default :
```

```
switch(2)
        {
          default : pf("Han Ho sakta hain Guru");
                    break;
          case 3 : pf("Maja Aagaya");
                   break;
        }
```

⑤ Default position can be anywhere.

```
int i=3;
switch(i)
        {
          case 1 : pf("one");
                   break;
          case 2 ; pf("Two");
                   break;
        }
```

compiler creates a table in memory corresponding to the cases

1	
2	

⑥ Default in a switch statement is optional.

```
int i=2;
switch(4)
    {
      i=i+2;          → never execute (ignored)
    case 2: pf("%d", i);
            break;
    case 4: pf("%d", i);
            break;
    default: pf("Hello");
    }
```

o/p : 2

1. statement written before all the cases will never execute

```
int i=2;
switch(i)
    {
    case 2: pf("%d", i);
            break;
            i=i+2;          → never executed (ignored)
    case 4: pf("%d", i);
            break;
    }
```

8. Only a code written b/w case label and break

```
case label : code
             break;
```

can execute

switch is not a loop

↳ continue → works on iterations (loops) cannot be used.

case printf( )

↓

we know function is an identifier

& case label ⇒ can only be a constant or constant expression (literal)

9. Mandatory part is the expression

we can have a switch like this as well

```
int a=2;
switch(a);
```

✓✓

10. No error but dummy switch

11. switch, case, default are keywords.

int a=2;
switch( );

compiler udd ke
laath Maarega.

in if (a>=1 && a<=10)
{
}
else if ( )
{
}

we can do this
for a range type
of values in
if

can we do this in switch

~~char ch = 'A';~~
~~switch (ch~~

int a=2;
switch (a) {

high value

case 1 ... 10 : pf (" Between 1 and 10 ");
Break;

low value — space — exactly 3 dots — space

}

***

12

Range can be provided in case by :

→ low value space ... space high value.

```
switch (z)
    {                    1, 2, 3, 4.
    case 1 ... 4 : pf ("Hello");
                   break;
    2, 3, 4, 5 ... 10
    case 2 ... 10 ; pf("Maja Aagaya");
                   break;
    }
```

Duplicate case label udd ke aaya

Q consider a non-leap year and we have to print days in a month.

```
switch(month) {
    case 1 :
    case 3 :
    case 5 :
    case 7 :
    case 8 :
    case 10 :
    case 12 : pf("31");
              break;
    case 2 : pf("28");
             break;
    case 4 :
    case 6 :
    case 9 :
    case 11 : pf("30");
              break;
}
```

```
switch (marks)
{
    case 1 ... 100 : switch(marks)
                     {
                         case 91 ... 100 : pf("A+");
                                           break;
                         case 81 ... 90  : pf("A");
                                           break;
                         case 71 ... 80  : pf("B+");
                                           break;
                         .
                         .
                         .
                     }
    default : pf("Arey yaar tu toh fail ni hoga");
}
```

13. For a group of value → same code will execute

14. Nesting of switch statements is also possible

Q
```
int i = 10;
switch(i)
{
    case 10 ... 1 : pf("sir ne nhi padhaya");
                    break;
    default       : pf("Bekaar");
}
```

Reverse range ⇒ Error

Q
```
switch (3)
{
    case 2 : printf ("Gate");
             break;
    case 2/1: pf (" Academy ");
             break;
    case 3 : pf("Hello");
}
```

2/1 = 010
001
011 ⇒ 3

duplicate case labels not allowed.

Q
```
switch (10)
    case 1 : pf("Hello");
             break ;
    case 2 : pf("Hello");
             break ;
```

error ⇒ cases not within switch
break not in a loop or switch

Q
```
switch(2)
{
    case 1:
        break;
}
```

NO O/P

Q
```
switch(2)
{
    default:
}
```

why error because

what is default ⇒ when nothing runs default runs

and what are we doing running a default statement and within it we are doing nothing which is contradictory ⇒ thats why error.

↳ its like using '%' operator on floating point values.

```
switch(2)
{
    default: ;
}
```
⇒ This is OK

```
switch(2)
{
    case 2 : int i=2;
             pf("%od", i);
             break;
}
```

Error: we cannot declare inside a switch

⇓ a label can only be a statement and a declaration is not a statement

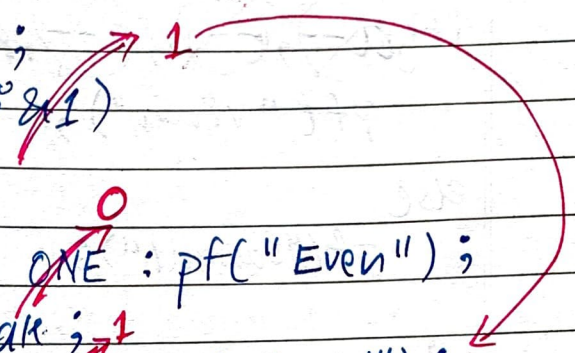
Q
```
switch(1)
{
    case 1 : {
                 int i=1;
                 pf("%od", i);
             }
}
```

{ } ⇒ specify a start of new scope.

and at a start of a new scope we can declare

WB
Q4

```
#define ONE 0
#define ZERO 1
int main()
{
    int i=3;
    switch(i&1)
    {
        case ONE : pf("Even");
        break;
        case ZERO : pf("odd");
        break;
        default : pf("Default");
    }
```



o/p : odd.

WBQ24, Q25 Practice Question 13 solved.