

CS1101: Foundations of Programming

Condition and Branching



Dept. of Computer Science & Engineering
Indian Institute of Technology Patna

Conditional statement

- Allow different sets of instructions to be executed depending on truth or falsity of a logical condition
- How do we specify these conditions?
 - Using expressions and relational operators (>,<,>=,<=,==,!=)
 - Non-zero value signifies condition is true
 - Value 0 indicates the condition is false
 - Using logical connectives (&&, ||, !)
- Example:

```
(a + b) <= 60  
((age == 18) && (height > 5))  
((grade == 'A') && ((bankbalance > 6000) || (age >= 45)))
```

Branching: if statement

```
if(expression)  
    statement;
```

```
if(expression){  
    Block of statements  
}
```

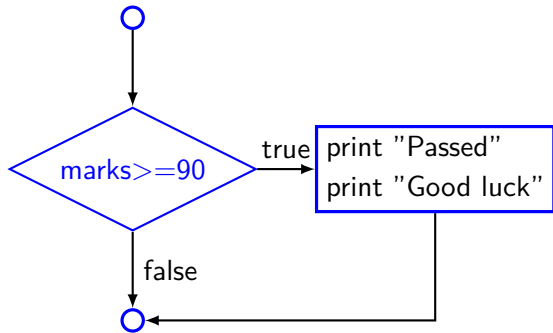
Indentation - (leaving horizontal spaces in the following lines of if) easy to read

The expression is evaluated and if it is non-zero, the statement gets executed. For 2nd case whole block of statements get executed.

Semi-colon may be noted in the first case.

Example

```
if(marks >= 90){  
    printf("Passed\n");  
    printf("Good luck\n");  
}
```



Branching: if-else statement

```
if(expression){  
    Block of statements;  
}  
else{  
    Block of statements;  
}
```

```
if(expression){  
    Block of statements;  
}  
else if(expression){  
    Block of statements;  
}  
else{  
    Block of statements;  
}
```

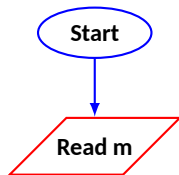
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```

Start

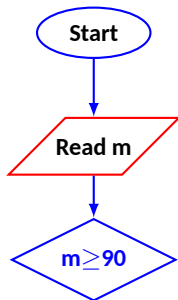
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



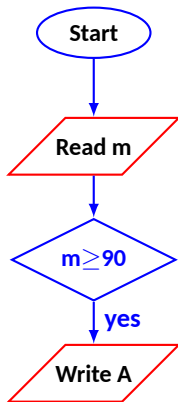
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



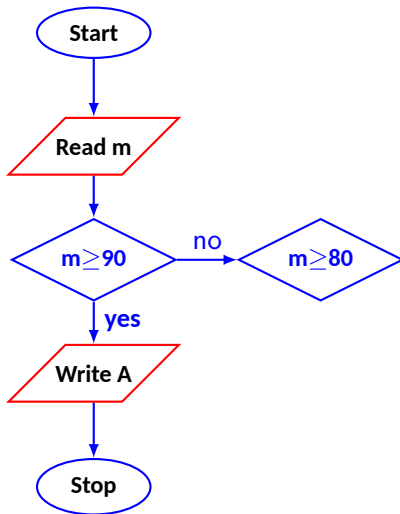
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



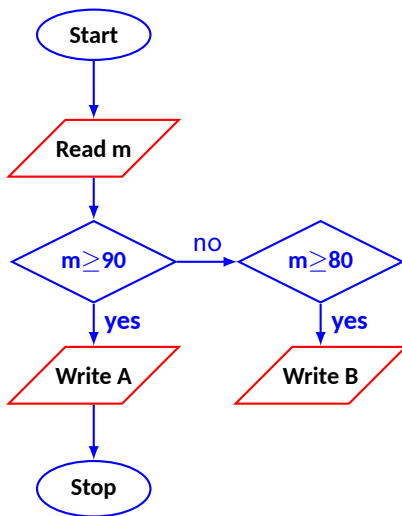
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



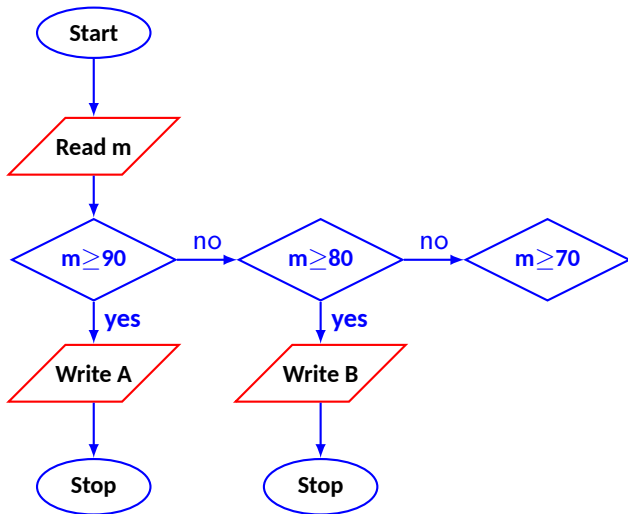
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



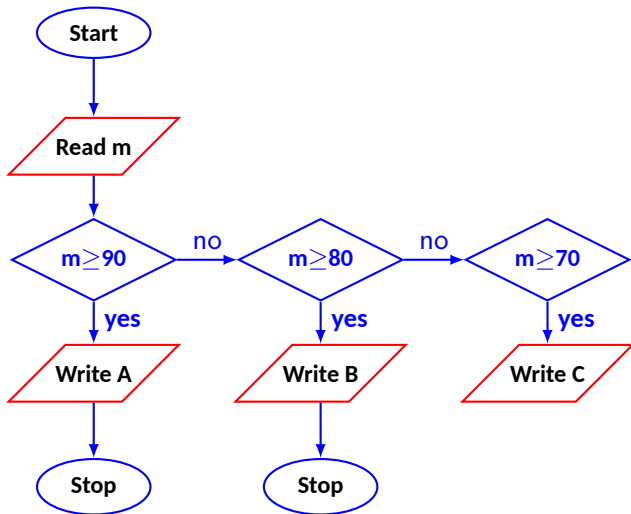
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



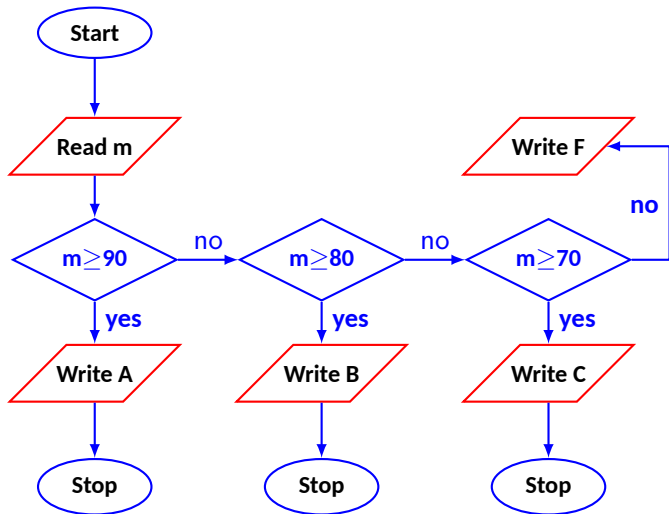
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



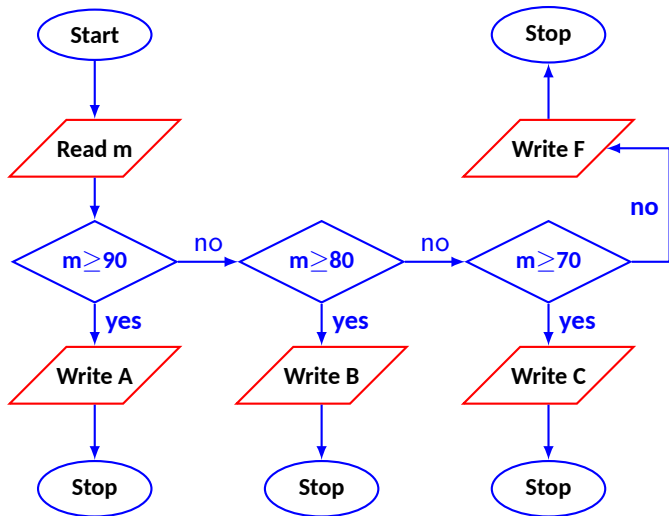
Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```



Example: grade computation

```
int main(){  
    int marks;  
    scanf("%d",&marks);  
    if(marks>=90)  
        printf("A");  
    else if(marks>=80)  
        printf("B");  
    else if(marks>=70)  
        printf("C");  
    else  
        printf("failed");  
    return 0;  
}
```

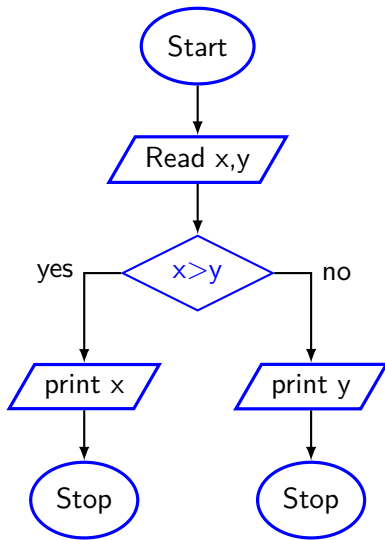


Example: grade computation

```
int main(){
    int marks;
    scanf("%d",&marks);
    if(marks>=90)
    { printf("A\n"); printf("Well done\n"); }
    else if(marks>=70)
        printf("B");
    else if(marks>=50)
        printf("C");
    else
    { printf("failed\n"); printf("Study hard\n"); }
    return 0;
}
```

Maximum of two numbers

```
#include <stdio.h>
int main()
{
    int x,y;
    scanf("%d%d",&x,&y);
    if(x>y)
        printf("Largest is %d\n",x);
    else
        printf("Largest is %d\n",y);
    return 0;
}
```

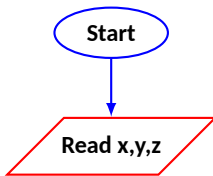


Example: minimum of 3 numbers

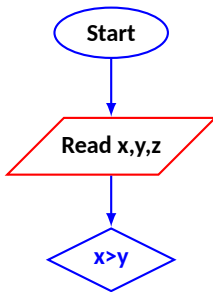


Start

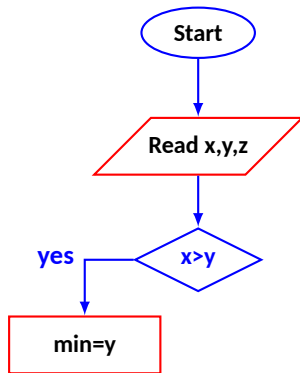
Example: minimum of 3 numbers



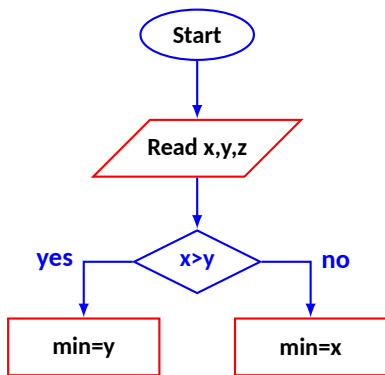
Example: minimum of 3 numbers



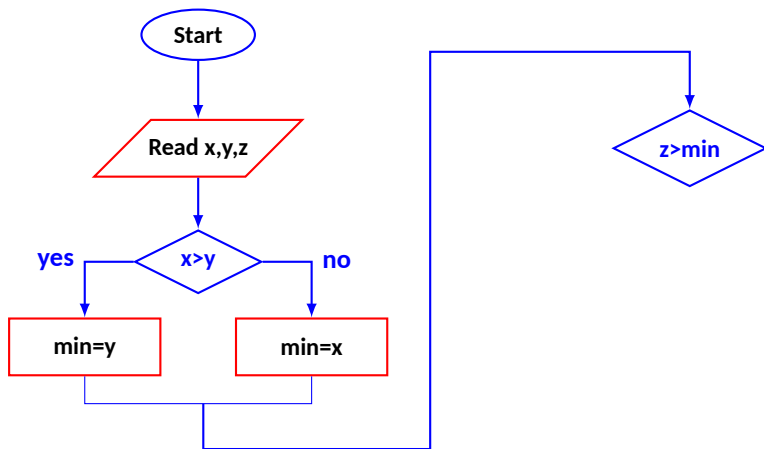
Example: minimum of 3 numbers



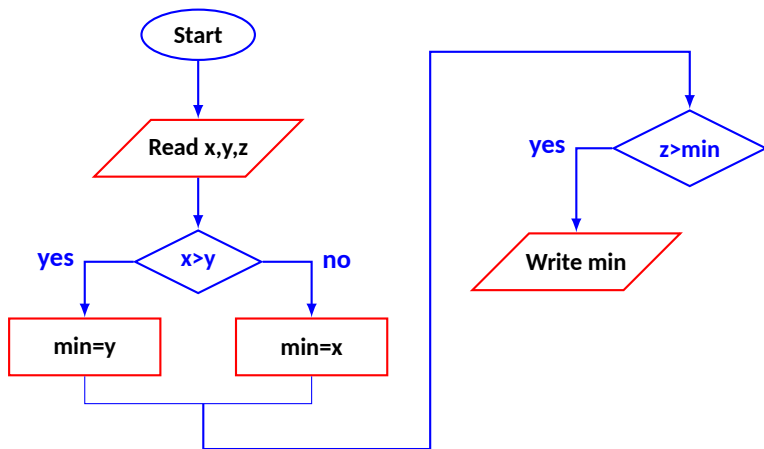
Example: minimum of 3 numbers



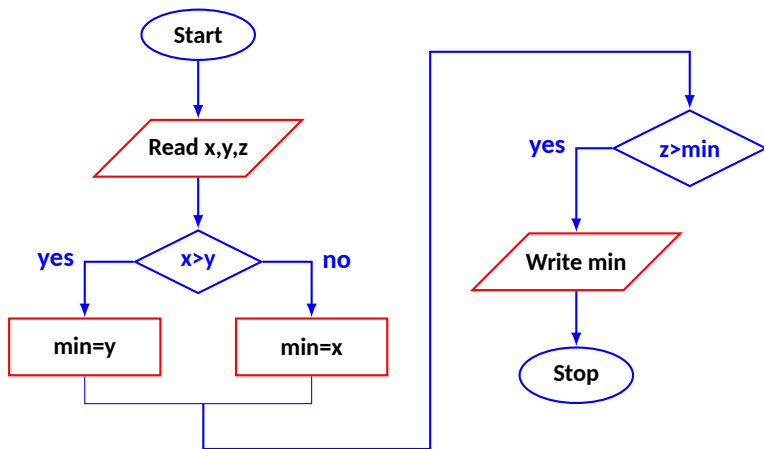
Example: minimum of 3 numbers



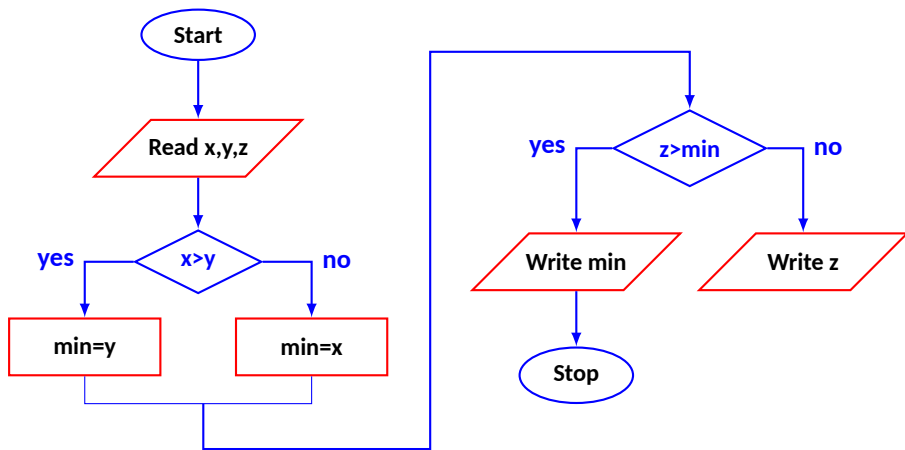
Example: minimum of 3 numbers



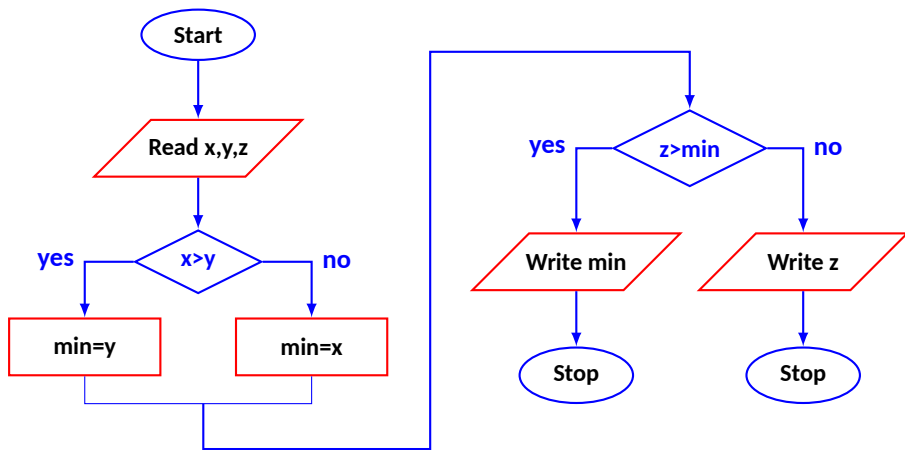
Example: minimum of 3 numbers



Example: minimum of 3 numbers



Example: minimum of 3 numbers



Minimum of three numbers

```
#include <stdio.h>
int main()
{
    int x,y,z,min;
    scanf("%d%d%d",&x,&y,&z);
    if(x > y) { min=y; }
    else { min=x;}
    if(z > min) { printf("%d",min); }
    else { printf("%d",z); }
    return 0;
}
```

Maximum of three numbers

```
int main(){
    int x,y,z;
    scanf("%d%d%d", &x, &y, &z);
    if((x >= y) && (x >= z))
        printf("%d",x);
    if((y >= x) && (y >= z))
        printf("%d",y);
    if((z >= y) && (z >= x))
        printf("%d",z);
    return 0;
}
```

Nesting of if-else structures

- It is possible to nest if-else statements, one within another
- All if statements may not be having the else part
 - Confusion??
- Rule to be remembered:
 - An "else" clause is associated with the closest preceding unmatched "if"

Dangling else problem

- `if(exp1) if(exp2) stmta else stmtb`

```
if(exp1){  
    if(exp2)  
        stmta  
    else  
        stmtb  
}
```

```
if(exp1){  
    if(exp2)  
        stmta  
}  
else  
    stmtb
```

While implementing a code, it is good practice to provide explicit { and } to avoid any confusion

Dangling else problem

- `if(exp1) if(exp2) stmta else stmtb`

```
if(exp1){  
    if(exp2)  
        stmta  
    else  
        stmtb  
}
```

```
if(exp1){  
    if(exp2)  
        stmta  
    }  
    else  
        stmtb  
    X
```

While implementing a code, it is good practice to provide explicit { and } to avoid any confusion

More examples

```
if e1 s1  
else if e2 s2
```

More examples

```
if e1 s1  
else if e2 s2
```

```
if e1 s1  
else {if e2 s2}
```

More examples

```
if e1 s1  
else if e2 s2
```

```
if e1 s1  
else {if e2 s2}
```

```
if e1 s1  
else if e2 s2  
else s3
```

More examples

```
if e1 s1  
else if e2 s2
```

```
if e1 s1  
else {if e2 s2}
```

```
if e1 s1  
else if e2 s2  
else s3
```

```
if e1 s1  
else {if e2 s2 else s3}
```

More examples

```
if e1 s1  
else if e2 s2
```

```
if e1 s1  
else {if e2 s2}
```

```
if e1 s1  
else if e2 s2  
else s3
```

```
if e1 s1  
else {if e2 s2 else s3}
```

```
if e1 if e2 s1  
else s2  
else s3
```

More examples

```
if e1 s1  
else if e2 s2
```

```
if e1 s1  
else {if e2 s2}
```

```
if e1 s1  
else if e2 s2  
else s3
```

```
if e1 s1  
else {if e2 s2 else s3}
```

```
if e1 if e2 s1  
else s2  
else s3
```

```
if e1 {if e2 s1 else s2}  
else s3
```

Example

- Print PASSED if a given marks is between 60 and 100, or FAILED if it is below 60. Do not print anything in other cases

```
int main(){
    int m;
    scanf("%d", &m);
    if(m >= 60)
        if(m <= 100)
            printf("PASSED\n");
    else
        printf("FAILED\n");
    return 0;
}
```

Example

- Print PASSED if a given marks is between 60 and 100, or FAILED if it is below 60. Do not print anything in other cases

```
int main(){
    int m;
    scanf("%d", &m);
    if(m >= 60)
        if(m <= 100)
            printf("PASSED\n");
    else
        printf("FAILED\n");
    return 0;
}
```

Output:

150
FAILED

Output:

30

Example

- Print PASSED if a given marks is between 60 and 100, or FAILED if it is below 60. Do not print anything in other cases

```
int main(){
    int m;
    scanf("%d", &m);
    if(m >= 60){
        if(m <= 100)
            printf("PASSED\n");
    } else
        printf("FAILED\n");
    return 0;
}
```

Output:

150

Output:

30

FAILED

Conditional operator ? :

- Conditional expression written with the ternary operator ? : provides alternate way to write if-else statement
- It is of the form — `expr1 ? expr2 : expr3 ;`
 - The expression `expr1` is evaluated first. If it is non-zero, then `expr2` is evaluated. Otherwise `expr3` will be evaluated
- Example: `z = (a > b) ? a : b ;`
`if (a > b)`
 `z = a;`
`else`
 `z = b;`
- Conditional expression can be represented in succinct way ? : operator
`printf("%c", (i%10==9 || i==n-1) ? '\n' : ' ');`

Equality (==) vs Assignment (=) operators

- It can lead to fatal error and does not cause syntax error in general
- Any expression that produces a value can be used for control structure
- Non-zero values are treated as true and zero as false
- Example:

```
if(age = 18)
    printf("You can cast vote!");
```

- `printf` statement will always get executed!!
- `if (18 = age)` — leads to syntax error, this can be good practice

The switch statement

- The `switch` statement is a multi-way decision that tests whether an expression matches one of a number of constant integer values and branches accordingly

```
switch (expr){  
    case cexpr1: S1  
    case cexpr2: S2  
    :  
    default: S  
}
```

- `expr` is any **integer** valued expression
- `cexpr1`, `cexpr2`, ... are **constant integer** valued expression and values must be distinct
- `S1`, `S2`, ..., `S` are statements or compound statements
- `default` is optional and can come anywhere not necessary at the end. Usually it is put at the end

The switch statement

```
switch (expr){  
    case cexpr1: S1  
    case cexpr2: S2  
    :  
    default: S  
}
```

- When a switch statement is executed, the expr is evaluated and control is transferred to the group of case statements
- It is then compared with cexpr1, cexpr2, ... for equality in order
- If there is a match, all statements from that point till the end of switch is executed (including default)
- Need to use break statement to transfer control to the end
- Statements for default gets executed if none of the expression is matched

Example

```
switch (choice){  
    case 'r':  
        printf("RED");  
        break;  
    case 'g':  
        printf("GREEN");  
        break;  
    default:  
        printf("OTHER");  
}
```

Output: (choice = 'r')

RED

Output: (choice = 'G')

OTHER

Example

```
switch (choice){  
    case 'r':  
    case 'R':  
        printf("RED");  
        break;  
    case 'g':  
    case 'G':  
        printf("GREEN");  
        break;  
    default:  
        printf("OTHER");  
}
```

Since no break statement is there, the control passes to the next statement without checking the next condition.

Output: (choice = 'r')

RED

Output: (choice = 'G')

GREEN

Output: (choice = 'z')

OTHER

Alternative way

```
switch (toupper(choice)){  
    case 'R':  
        printf("RED");  
        break;  
    case 'G':  
        printf("GREEN");  
        break;  
    default:  
        printf("OTHER");  
}
```

Output: (choice = 'r')

RED

Output: (choice = 'G')

GREEN

Output: (choice = 'z')

OTHER

The break statement

- Used to exit from a switch or terminate from a loop
- With respect to switch, the break statement causes a transfer of control out of the entire switch statement, to the first statement following the switch statement
- Can be used with other statements also

Example

```
int x;  
scanf("%d",&x);  
switch(x){  
    case 1: printf("One \n");  
    case 2: printf("Two \n");  
    default: printf("Not one or two \n");  
}
```

- Let entered value is 1, output will be

Example

```
int x;  
scanf("%d",&x);  
switch(x){  
    case 1: printf("One \n");  
    case 2: printf("Two \n");  
    default: printf("Not one or two \n");  
}
```

- Let entered value is 1, output will be

One

Two

Not one or two

Example: Find 2nd max of 3 numbers

```
int main(){
    int x, y, z, fmax, smax;
    scanf("%d%d%d",&x,&y,&z);
    if(x > y) { fmax = x; smax = y;}
    else { fmax = y; smax = x;}
    if(z > fmax) { smax = fmax;}
    else if(z > smax) { smax = z;}
    else { }
    printf("Second max = %d\n", smax);
    return 0;
}
```

Practice problems

- Read in 4 integers and print the second maximum number
- Read in the coefficient a , b , c of the expression $ax^2 + bx + c = 0$. Print the roots of the equation nicely (For imaginary roots, use $x + iy$ format)
- Read in 3 points on a 2D plane and check if they are collinear. Print suitable message
- Read in a number (integer), convert it into grade. Marks ≥ 90 is AA, 80-89 — AB, 70-79 — BB, 60-69 — BC, 50-59 — CC, 40-49 — CD, 30-39 — DD and marks < 30 will get F