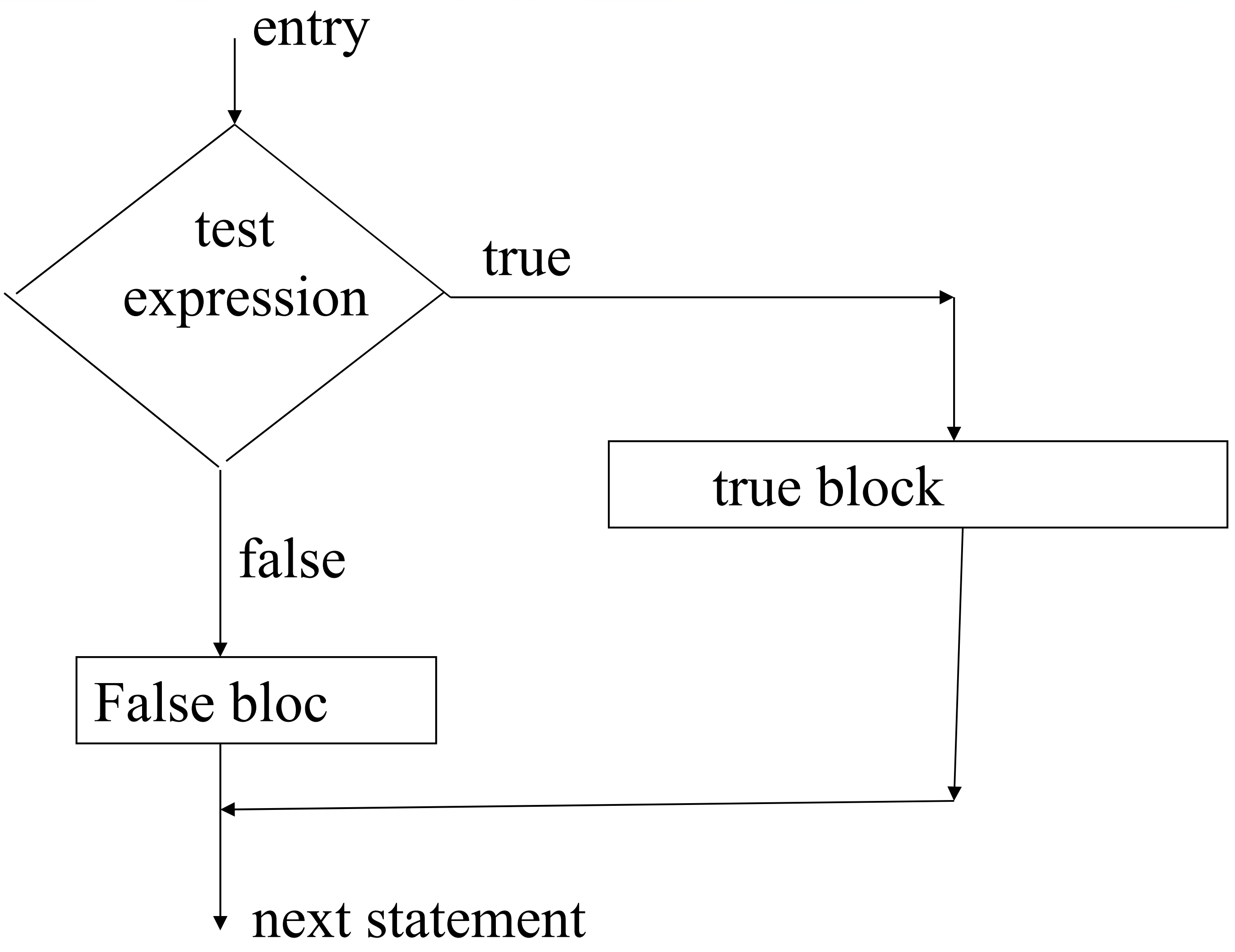
C

DECISSIONS !

CONTROL STATEMENTS

- ✓**if** statement
- ✓**switch** statement
- ✓**conditional operator** statement
- ✓**goto** statement

if statement



entry

test expression

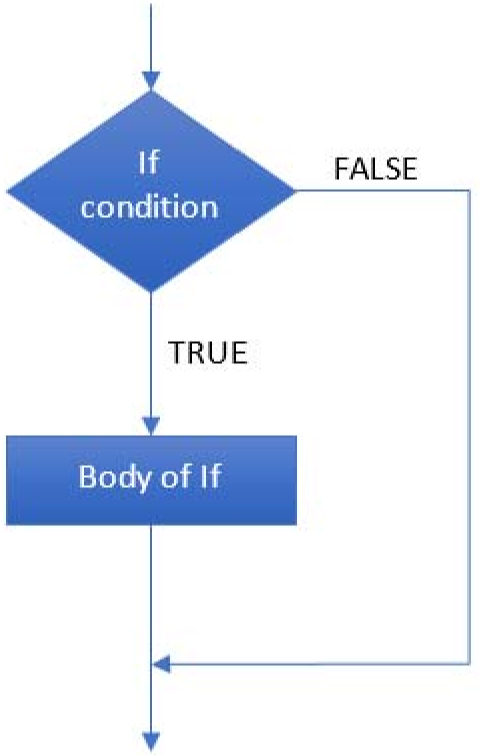
true

true block

false

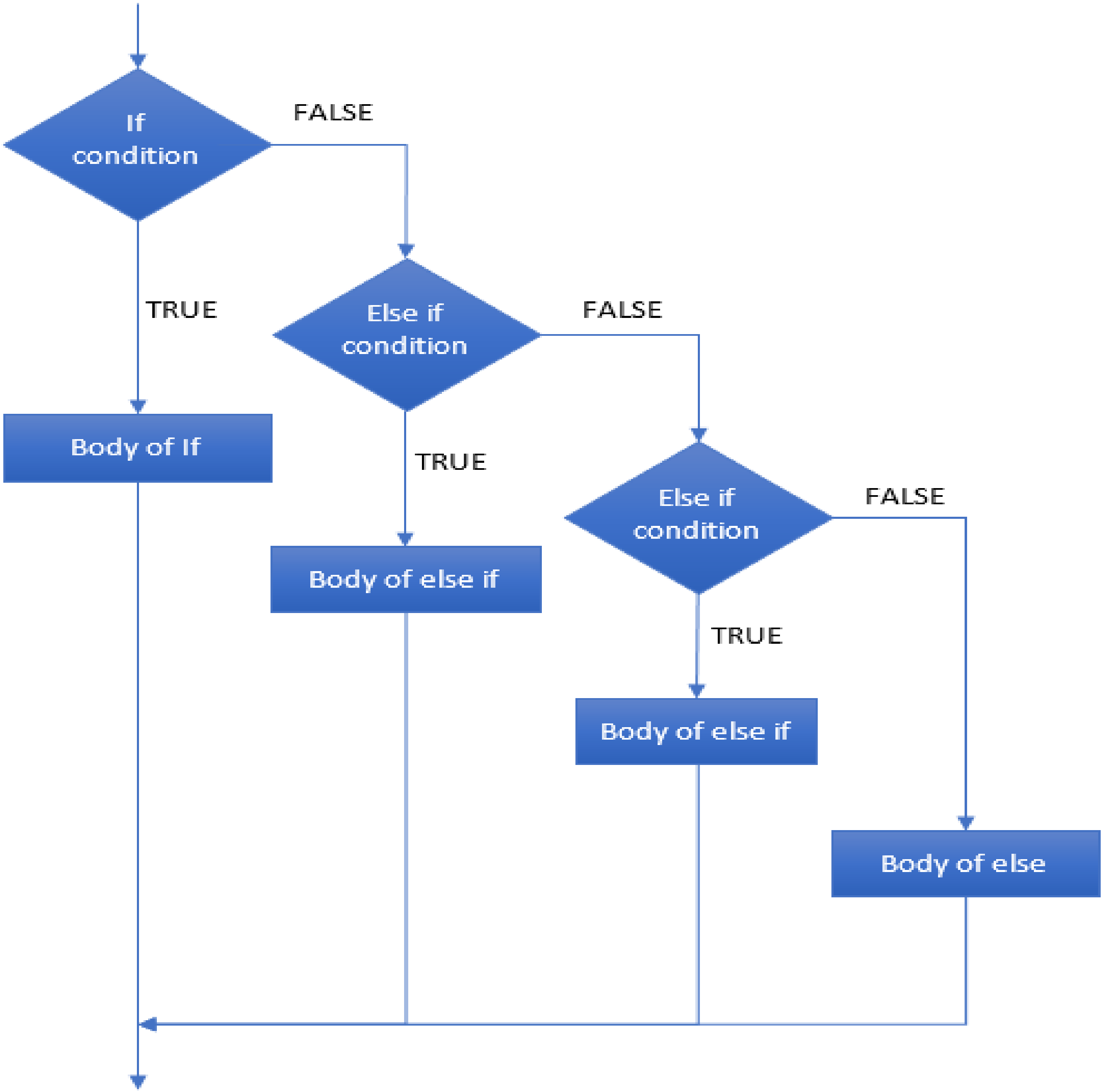
False bloc

next statement

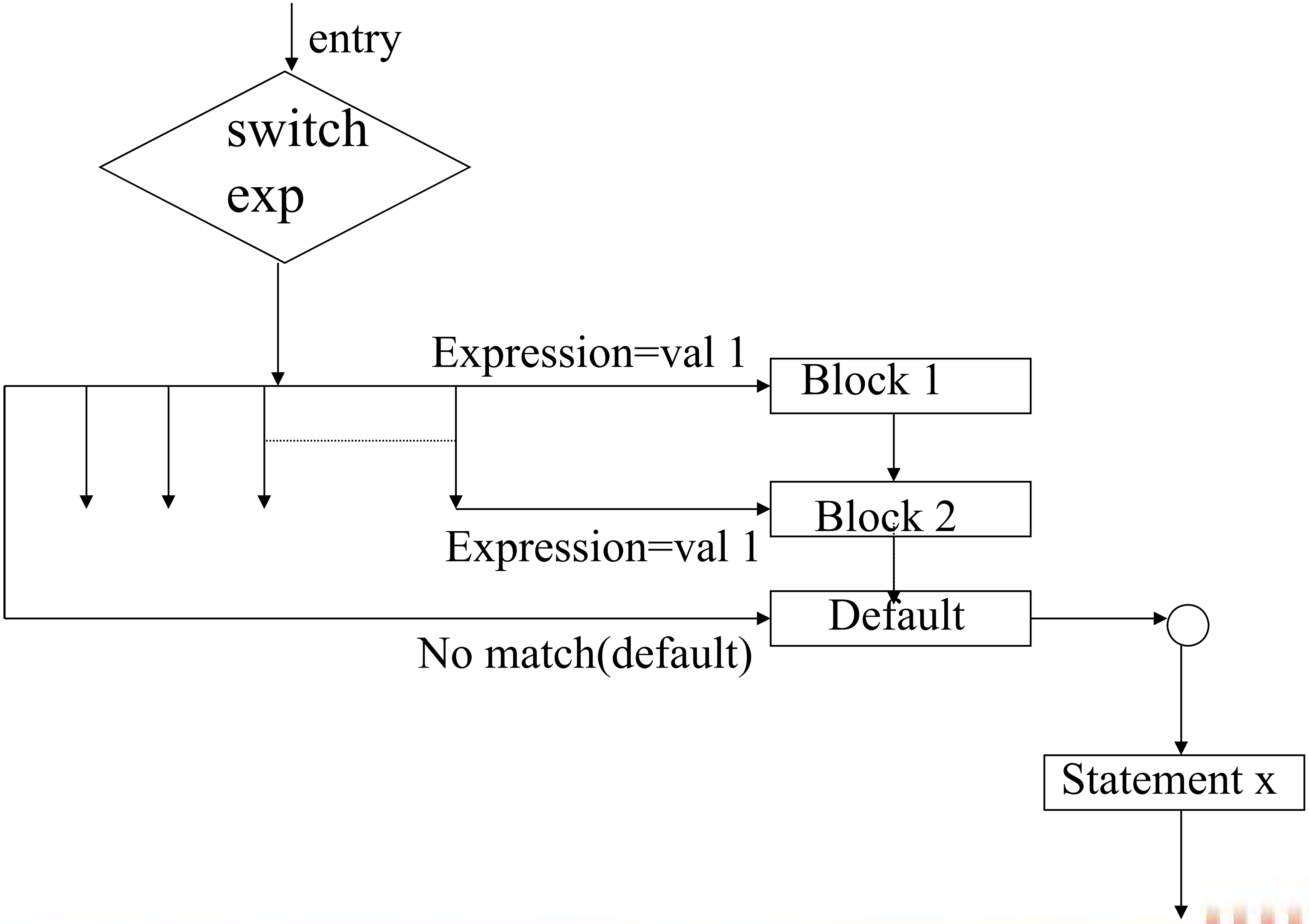

If condition
FALSE
TRUE
Body of If

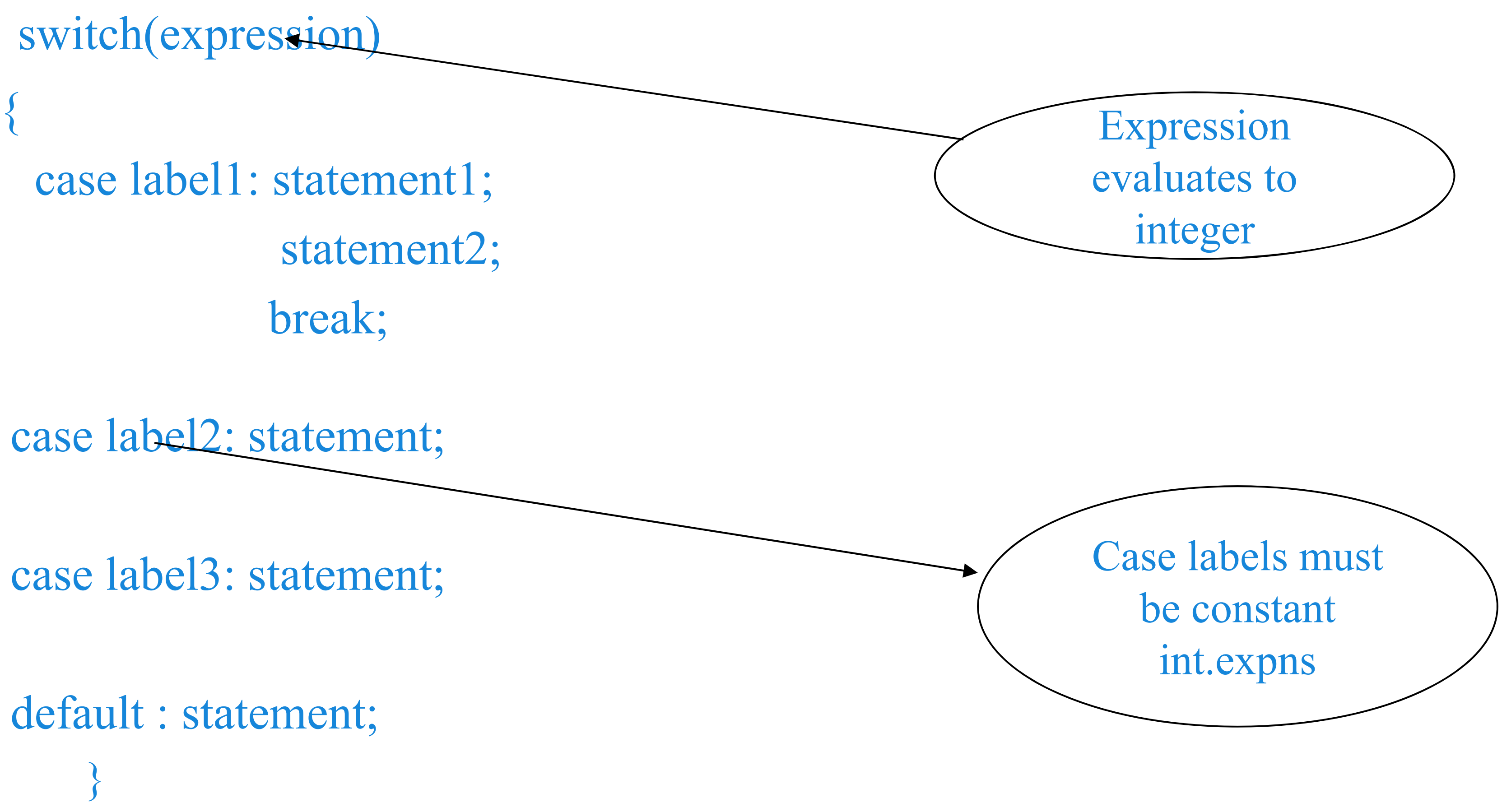
if……..else statement (optional else)

```
if(test expression)
        True block statement;
else
        false block statement;
statement x;
```


If condition
FALSE
TRUE
Else if condition
FALSE
TRUE
Body of If
Else if condition
FALSE
Body of else if
TRUE
Body of else if
Body of else

switch STATEMENT



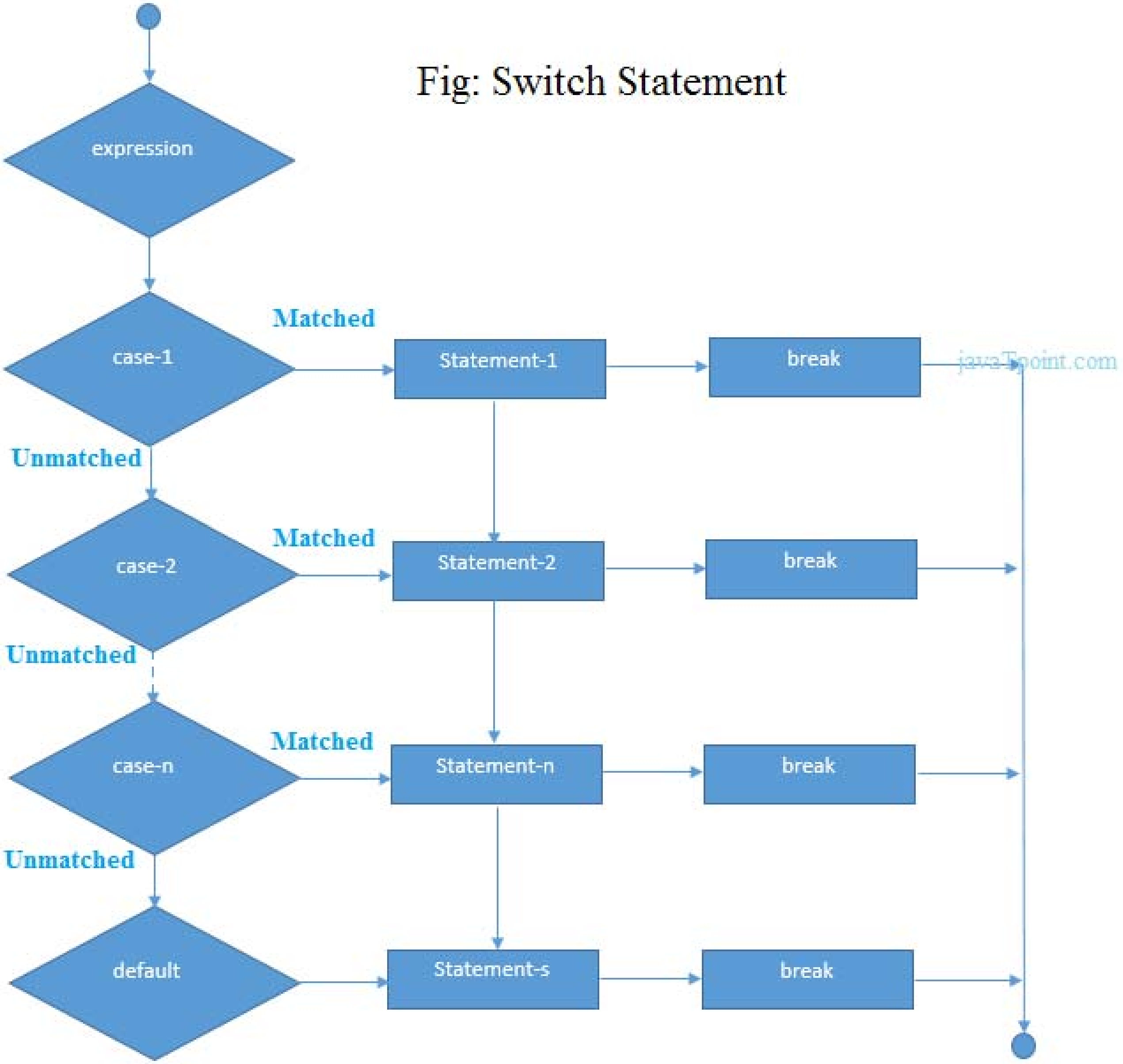


```
 switch(expression)
{
  case label1: statement1;
                      statement2;
                     break;

case label2: statement;

case label3: statement;

default : statement;
      }
```

Expression evaluates to integer

Case labels must be constant int.expns

Fig: Switch Statement



THE ? : OPERATOR

CONDITIONAL EXPRESSION? EXPRESSON1:EXPRESSION2

Ex:

```
if(x<0)
        flag=0;
else
        flag=1;
```

Same statements using ? : operator

```
flag= (x < 0) ? 0 : 1;
```

THE GOTO STATEMENT

```
        goto label;
        statement1;
        statement2;

        statementn;
label:  statement;
```

DECISION MAKING AND LOOPING

while loop is an entry controlled loop

```
while(test condition)
     {
          body of the loop
     }
```

```
sum=0;
n=1;
while(n<=5)
{       sum=sum+n;
        n=n+1;
}
printf("sum=%d",sum);
```

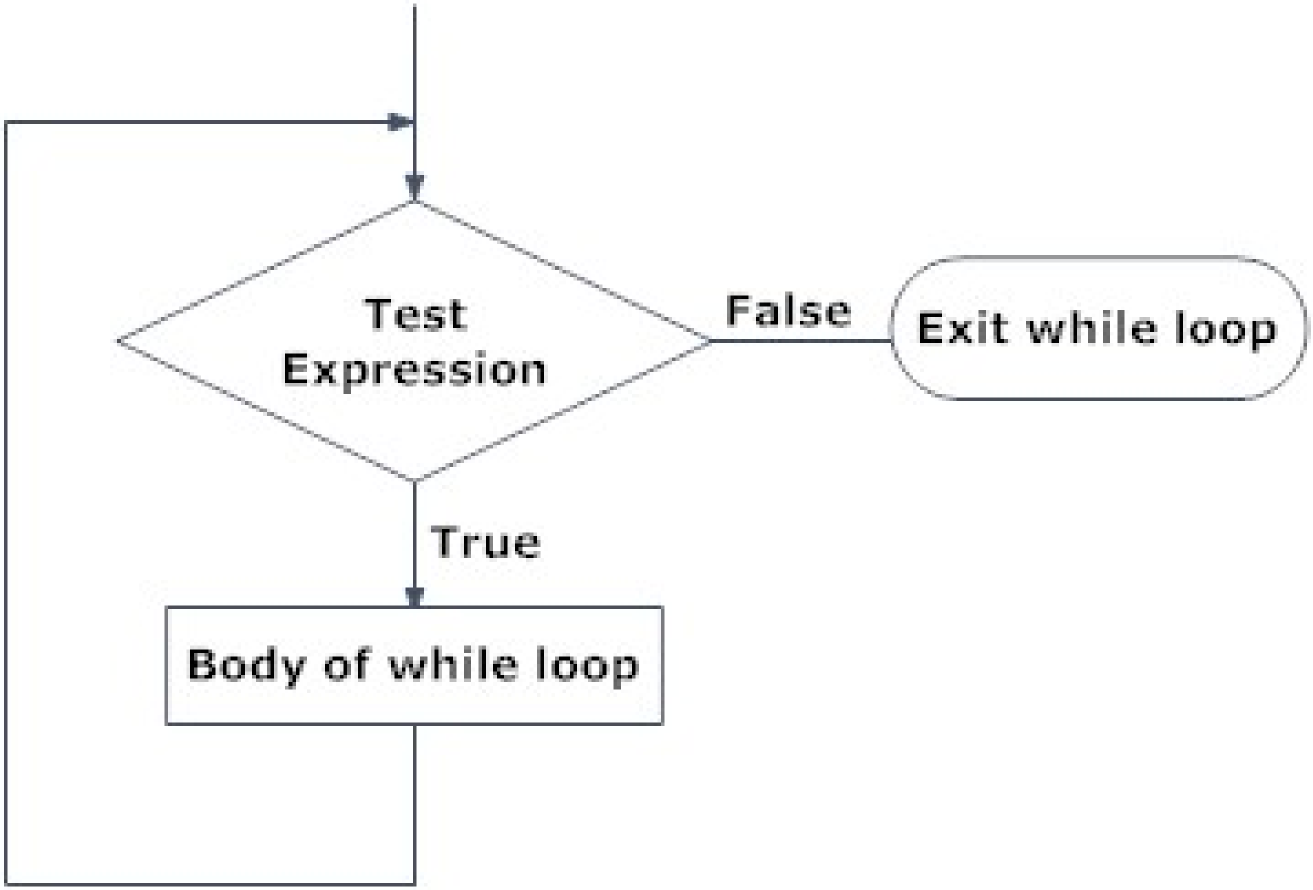


Figure: Flowchart of while loop

THE do-while LOOP

the do-while loop

```
do
{
        body of the loop
}
while(test condition);
```

is an exit controlled statement

Executed at least once

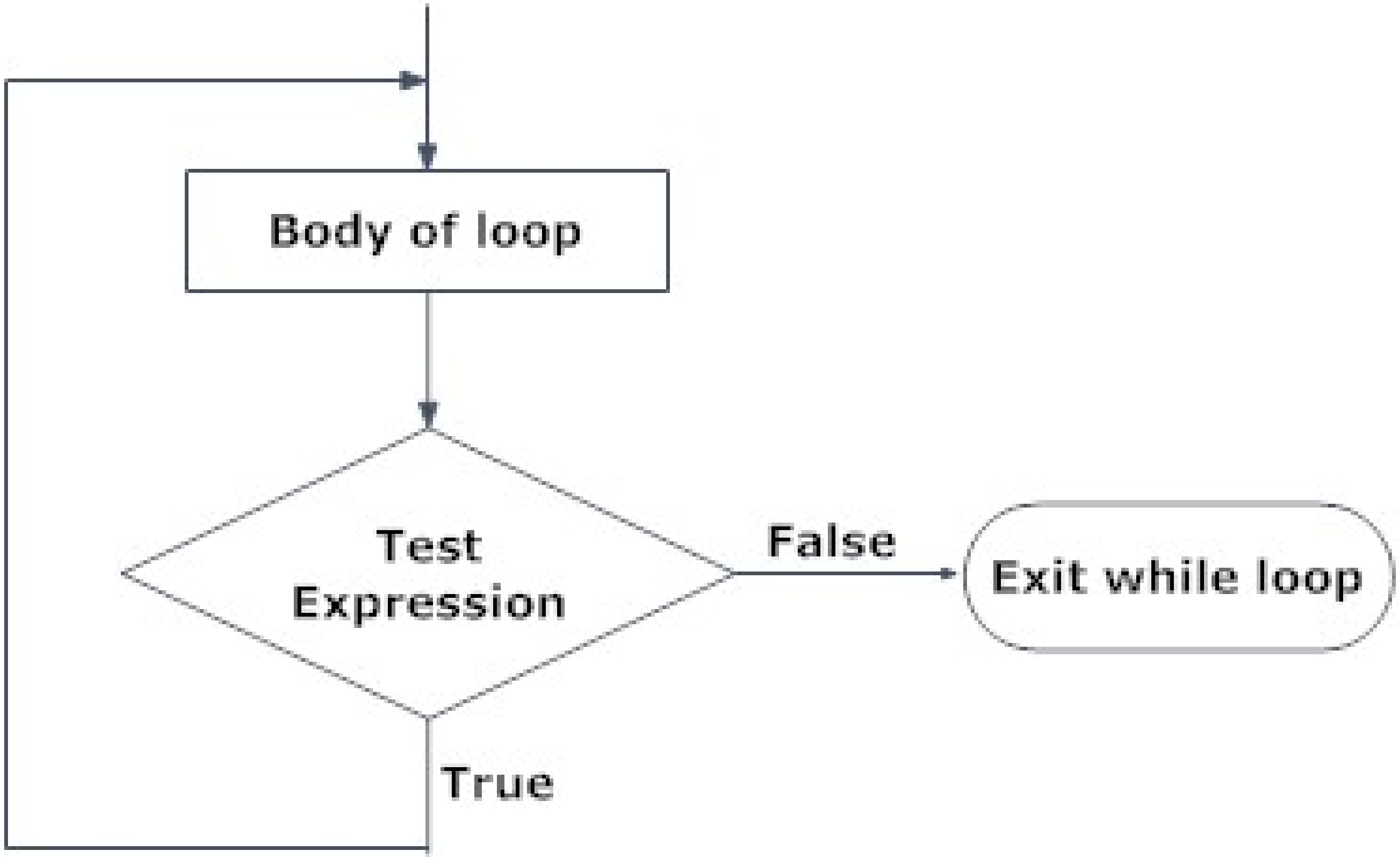


Figure: Flowchart of do...while loop

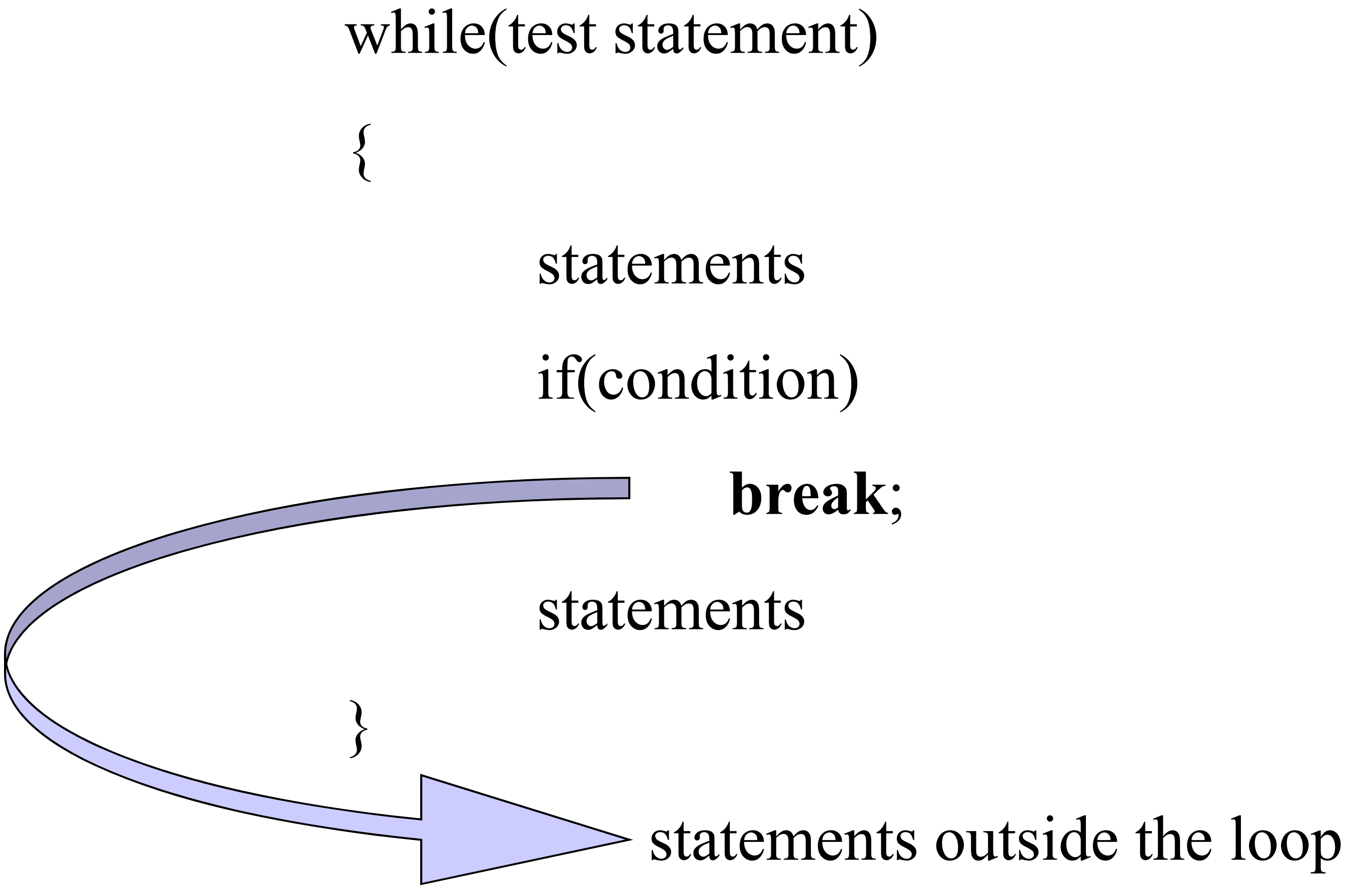
THE for LOOP

```
for (initialization ; test condition ; increment)
{
        body of the loop
}
```

```
sum=0;

for ( i=0 ; i<6; i++)

{

        sum=sum+i;

}
```

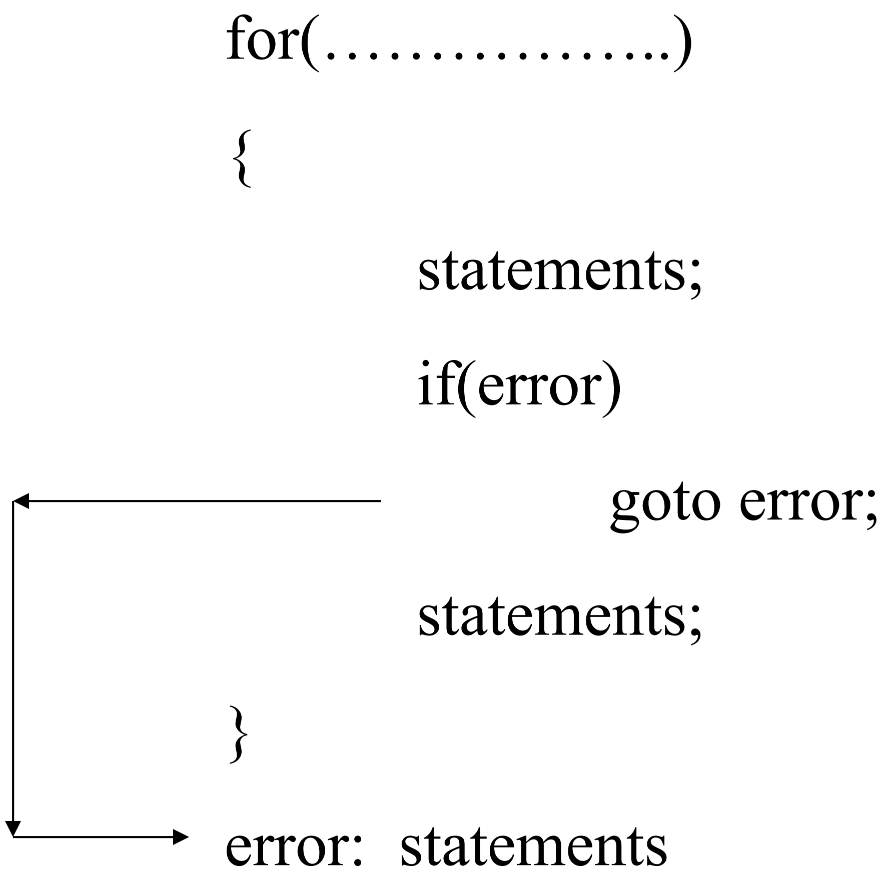
JUMPS IN THE LOOP

break statement



```
while(test statement)
{
        statements
        if(condition)
            break;
        statements
}
           statements outside the loop
```

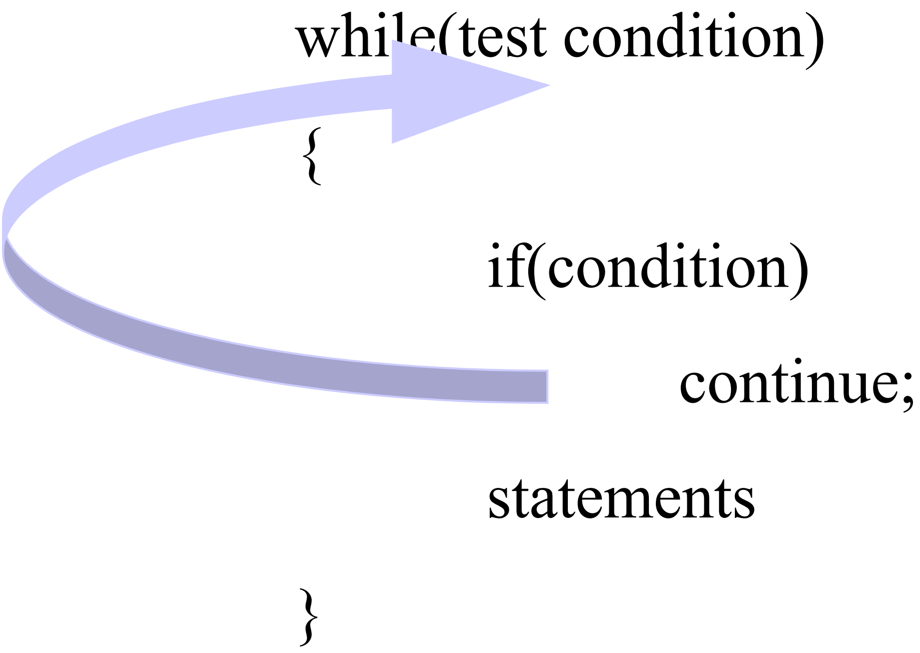
JUMP USING GOTO



```
for(……………..)
{
        statements;
        if(error)
                goto error;
        statements;
}
error:  statements
```

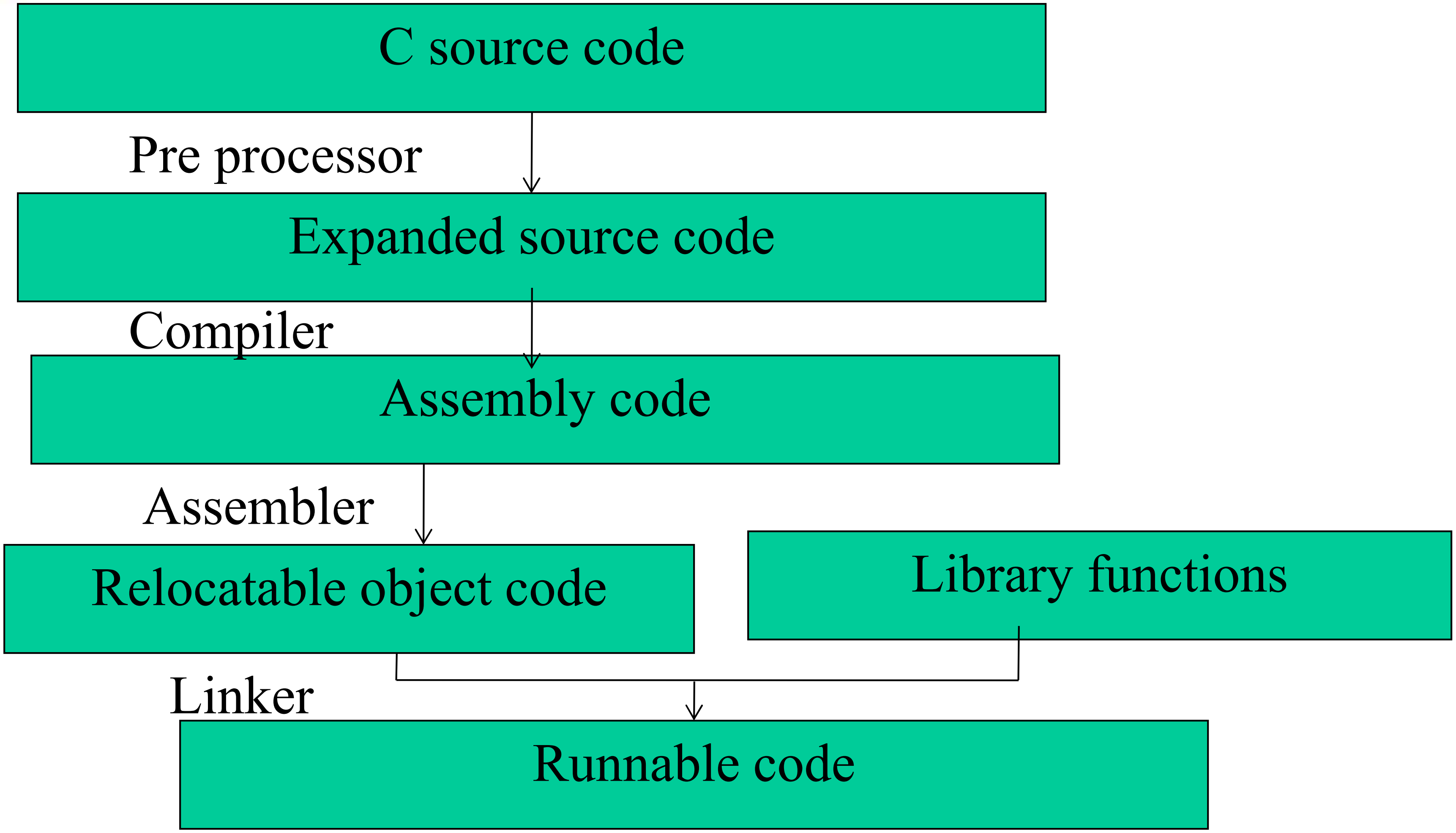
Skipping the current iteration

continue



```
while(test condition)
{
        if(condition)
                continue;
        statements
}
```

C Compilation process



C source code

Pre processor

Expanded source code

Compiler

Assembly code

Assembler

Relocatable object code

Library functions

Linker

Runnable code