AAB University

Faculty of Computer Sciences

Programimi i Orientuar në Objekte

Java 5:

Gjendjet e Kontrollit

Doc. Dr. **Mentor Hamiti**

mentor.hamiti@universitetiaab.com

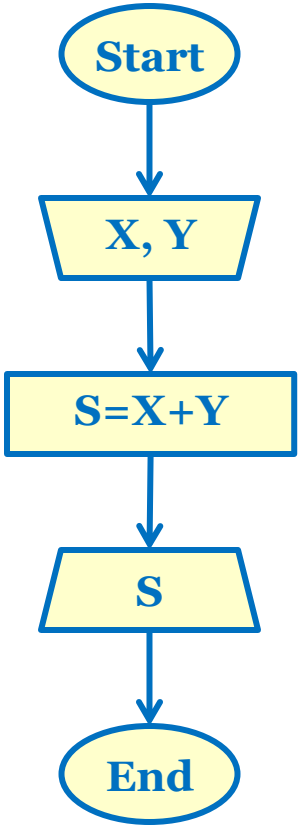
Hyrje

- Para se të fillohet me shkruarjen e programit, gjithmonë është e nevojshme që të kuptohet problemi dhe me kujdes të planifikohet qasja ndaj zgjidhjes së të njejtit

- Gjithashtu duhet të planifikohet edhe struktura e shkruarjes së kodit dhe teknikat e programimit në zbatim

Algoritmet (Algorithms)

- **Algoritmi** paraqet procedurën e zgjidhjes së problemit nga këndvështrimi i
  - Operacioneve të ekzekutuara dhe
  - Rradha e ekzekutimit të operacioneve
- Specifikimi i rradhës së operacioneve në kuadër të ekzekutimit të një programi kompjuterik njihet si kontroll programor (**program control**)
- Kontrolli programor realizohet në C++ përmes gjendjeve të kontrollit



Pseudokodi (Pseudocode)

- Pseudokodi është gjuhë artificiale dhe joformale që lehtëson zhvillimin dhe interpretimin e algoritmeve
- Pseudokodi i shkruar mirë lehtë mund të konvertohet në programin C++ përkatës
- Shembull:

```
1  Prompt the user to enter the first integer
2  Input the first integer
3
4  Prompt the user to enter the second integer
5  Input the second integer
6
7  Add first integer and second integer, store result
8  Display result
```

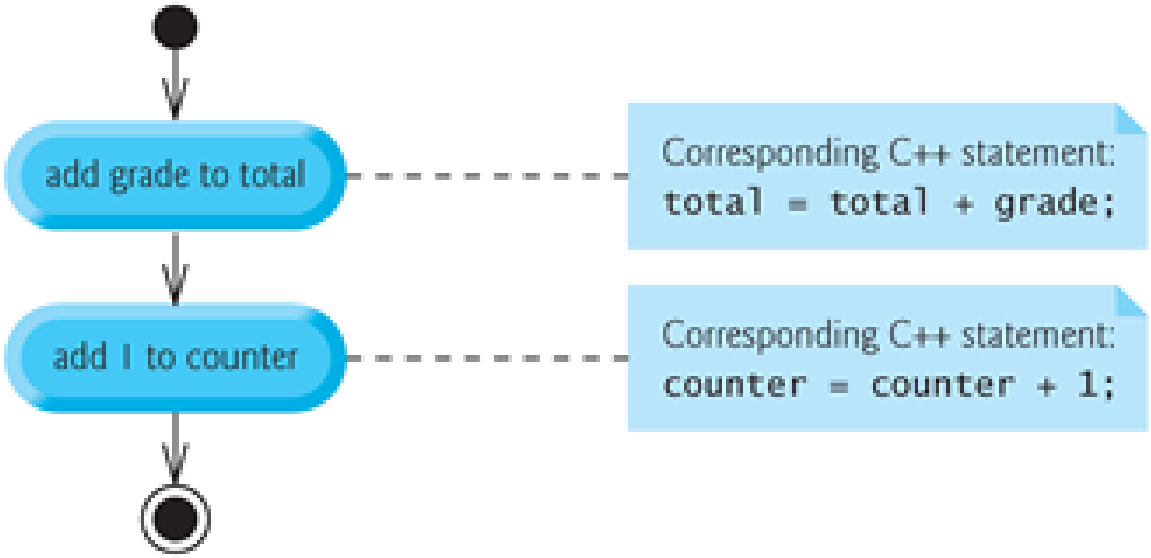
- Gjendjet e kontrollit

Struktura e Kontrollit

- Normalisht, urdhërat në kuadër të një programi ekzekutohen njëri pas tjetri, sipas rradhës së shkruar - **Mënyra sekuenciale**
- Në C++ egzistojnë urdhëra që kanë mundësi të ndryshojnë rradhën e ekzekutimit – **transferi I kontrollit**
- Të gjitha programet mund të shkruhen përmes tri llojeve të strukturave vijuese të kontrollit::
  - Sekuenciale
  - E zgjedhur (kushtëzuar)
  - Iterative (unazore)

Struktura e Kontrollit

- UML-diagrami (*UML- The **U**nified **M**odeling **L**anguage*):



  - *UML-diagramet kontribojnë në zhvillimin dhe interpretimin e algoritmeve, edhepse shumica e programerëve preferojnë **pseudocodet***

Struktura e Kontrollit - Selektimi

- C++ ofron tri mënyra të selektimit:
  - Me **if** ofrohet **selektimi i njëfishtë** ngase verifikohet plotësimi i një kushti të vetëm për egzekutimin ose mosegzekutimin e urdhërit në fjalë
  - Me **if...else** ofrohet **selektimi i dyfishtë** ngase zgjedhet njëri nga dy opcione varësisht nga plotësimi ose mosplotësimi i kushtit të definuar
  - Me **switch** ofrohet **zgjedhja e shumëfishtë** ngase nga më tepër mundësi të ofruara, zgjidhet njëra

Struktura e Kontrollit - Iteracioni

- C++ ofron tri mënyra të përsëritjes së gjendjeve (unazave) përderisa kushti i paraparë vazhdon të jetë i plotësuar

- Këto janë **while**, **do…while** dhe **for**

  - **while** dhe **for** ofrojnë përsëritjen e gjendjeve zero ose më tepër herë

  - **do…while** ofrojnë përsëritjen e gjendjeve së paku një herë

Struktura e Kontrollit - Iteracioni

- Çdo program në C++ kompozohet nga 7 versionet vijuese të strukturave të kontrollit:
  - Sekuenca
  - if
  - if ... else
  - switch
  - while
  - do ... while
  - for

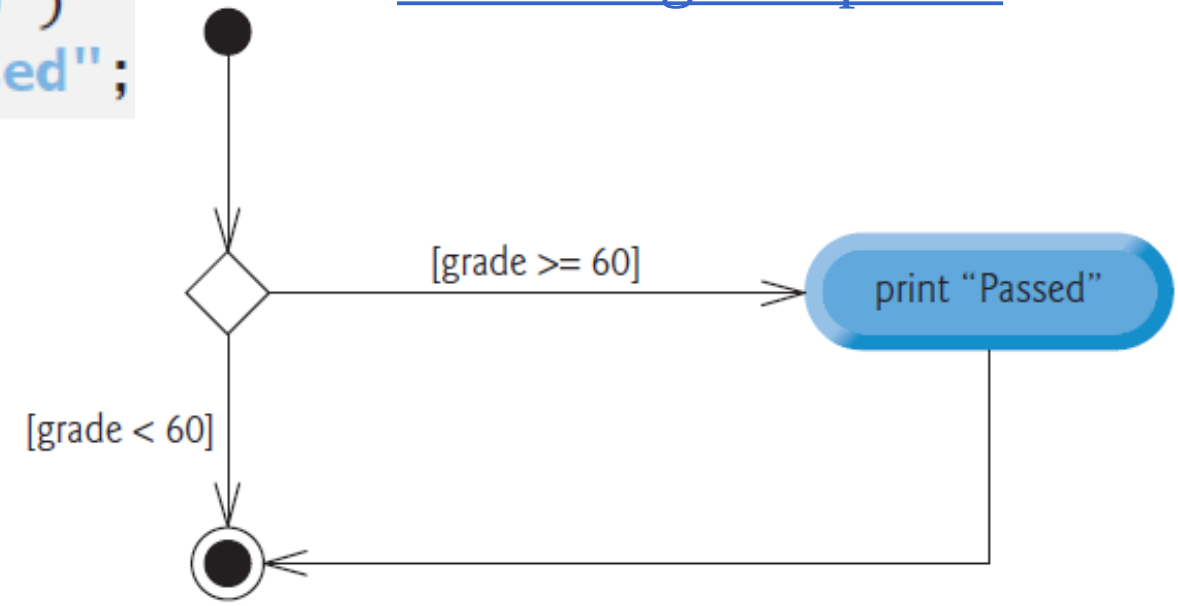
- Pseudokodi për **if**:

If student's grade is greater than or equal to 60
Print "Passed"

- C++ kodi për **if** :

```
if ( grade >= 60 )
   cout << "Passed";
```

UML-diagrami për **if** :



if … else

- **if…else** mundëson ekzekutimin e një urdhëri (ose grupi të urdhërave) kur plotësohet kushti, kurse kur nuk plotësohet ofrohet egzekutimi i urdhërit (grupit) tjetër

- Pseudokodi:

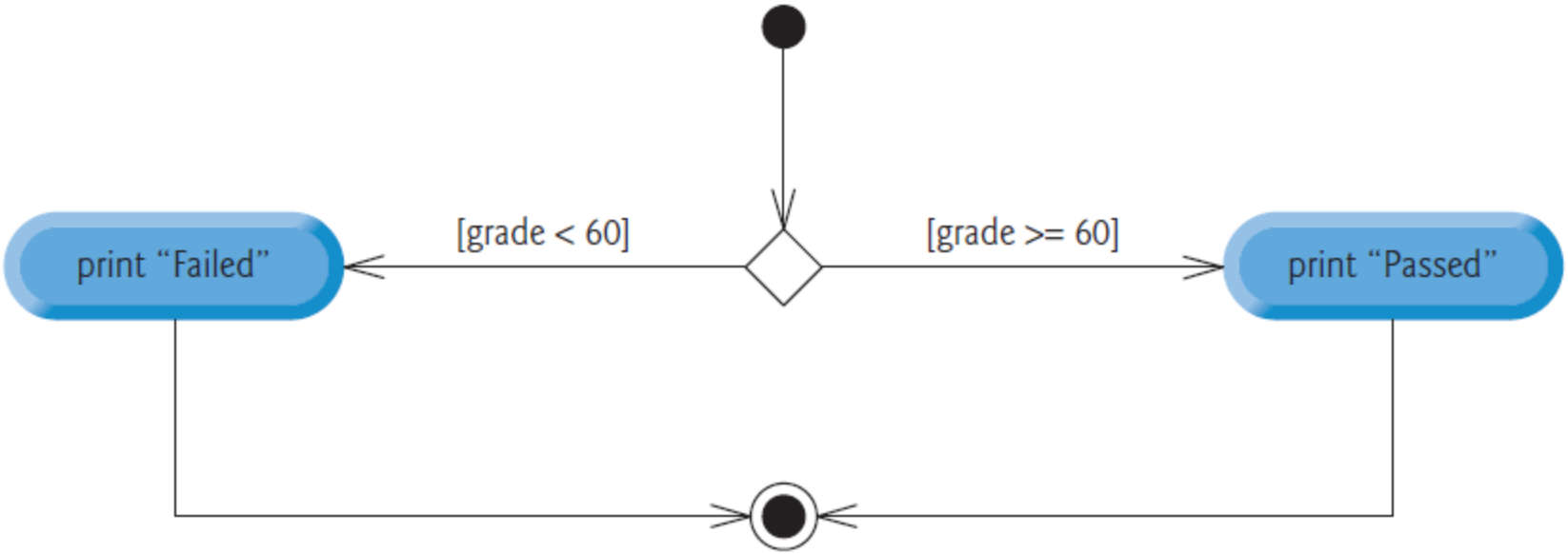
```
If student's grade is greater than or equal to 60
    Print "Passed"
Else
    Print "Failed"
```

- C++ kodi:

```
if ( grade >= 60 )
   cout << "Passed";
else
   cout << "Failed";
```

if ... else

- **if...else** UML-diagrami:



if ... else të ndërthurura

- Strukturat **if...else** mundësojnë zgjedhje të shumëfishtë në rastet kur përdoren njëra në kuadër të tjetrës
- Pseudokodi:

```
If student's grade is greater than or equal to 90
    Print "A"
Else
    If student's grade is greater than or equal to 80
        Print "B"
    Else
        If student's grade is greater than or equal to 70
            Print "C"
        Else
            If student's grade is greater than or equal to 60
                Print "D"
            Else
                Print "F"
```

if ... else të ndërthurura

- Pseudokodi mund të shkruhet në C++ si vijon:

```
if ( studentGrade >= 90 ) // 90 and above gets "A"
   cout << "A";
else
   if ( studentGrade >= 80 ) // 80-89 gets "B"
      cout << "B";
   else
      if ( studentGrade >= 70 ) // 70-79 gets "C"
         cout << "C";
      else
         if ( studentGrade >= 60 ) // 60-69 gets "D"
            cout << "D";
         else // less than 60 gets "F"
            cout << "F";
```

if ... else të ndërthurura

- Shumica e programerëve të njejtin kod e shkruajnë:

```
if ( studentGrade >= 90 ) // 90 and above gets "A"
   cout << "A";
else if ( studentGrade >= 80 ) // 80-89 gets "B"
   cout << "B";
else if ( studentGrade >= 70 ) // 70-79 gets "C"
   cout << "C";
else if ( studentGrade >= 60 ) // 60-69 gets "D"
   cout << "D";
else // less than 60 gets "F"
   cout << "F";
```

- *Të dy versinet janë plotësisht identike, përpos hapsirës së lirë e cila injorohet nga kompajleri!*

Dyshimet në ***else***

- Dyshimet në rrjedhën e ekzekutimit **else** mund të eliminohen me përdorimin e kllapave të mëdha **{** dhe **}**

```
if ( x > 5 )
   if ( y > 5 )
      cout << "x and y are > 5";
else
   cout << "x is <= 5";
```

```
if ( x > 5 )
{
   if ( y > 5 )
      cout << "x and y are > 5";
}
else
   cout << "x is <= 5";
```

while iteracioni

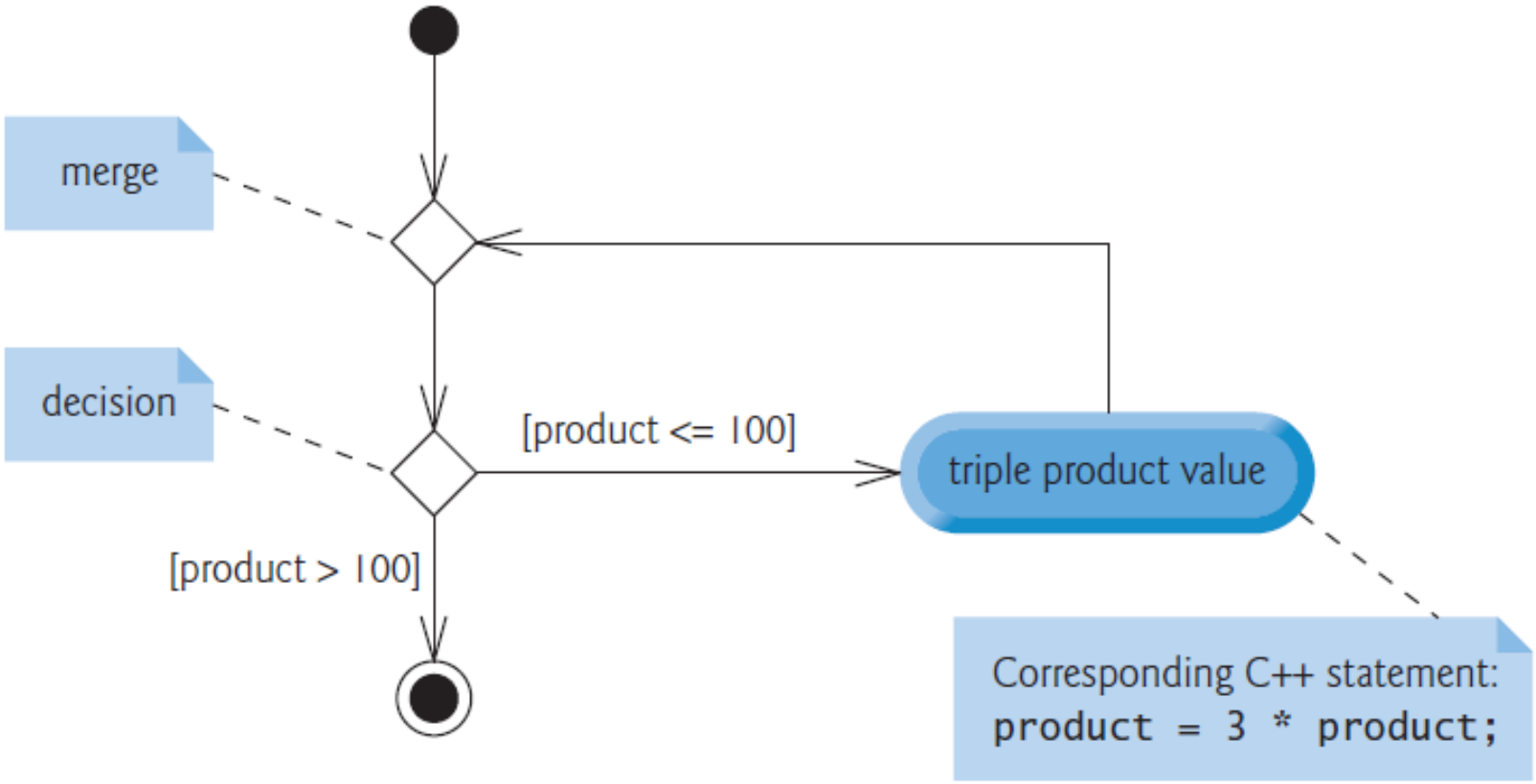
- Përsëritjet (unazat) lejojnë mundësinë e ekzekutimit të urdhërave të njejtë, përderisa vazhdon të ngelë i plotësuar kushti përkatës i definuar
- Shembull:
  - Të gjendet fuqia e numrit 3, e cila nuk do ta tejkalon vlerën 100!

```
int product = 3;

while ( product <= 100 )
   product = 3 * product;
```

  - *Përfundimi i kësaj unaze do ta përckton vlerën më ta lartë të prodhimit 3 me 3, e cila nuk do ta kalon 100!*

while iteracioni

- UML diagrami për interacionin **while** :



while iteracioni

- Shembull: Numërimi i kontrolluar

```
// Counter-controlled repetition.
#include <iostream>
using namespace std;

int main()
{
   int counter = 1; // declare and initialize control variable

   while ( counter <= 10 ) // loop-continuation condition
   {
      cout << counter << " ";
      ++counter; // increment control variable by 1
   } // end while

   cout << endl; // output a newline
} // end main
```

```
1 2 3 4 5 6 7 8 9 10
```

for iteracioni

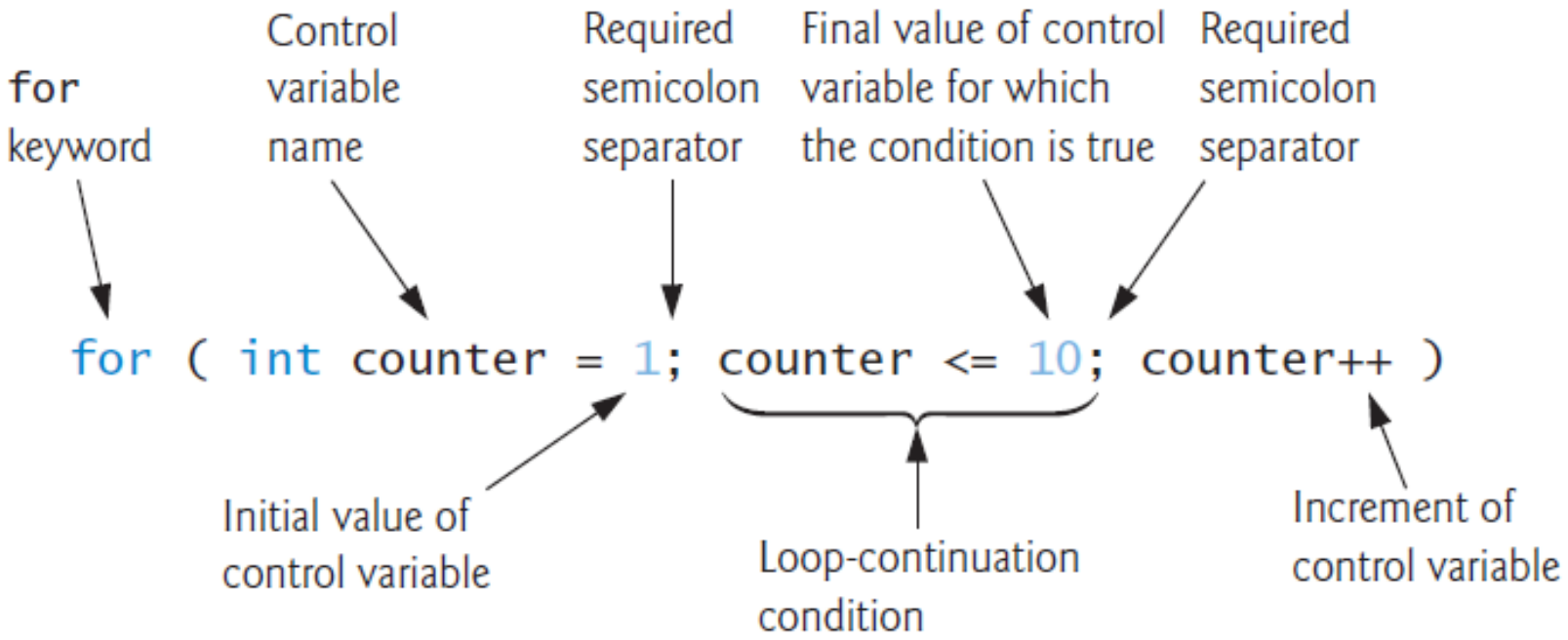
- Numërimi i kontrolluar përmes **for**

```
// Counter-controlled repetition with the for statement
#include <iostream>
using namespace std;

int main()
{
   // for statement header includes initialization,
   // loop-continuation condition and increment.
   for ( int counter = 1; counter <= 10; ++counter )
      cout << counter << " ";

   cout << endl; // output a newline
} // end main
```

```
1 2 3 4 5 6 7 8 9 10
```

for iteracioni

- **for** komponentet përbërëse:



for iteracioni

- **for ose while**

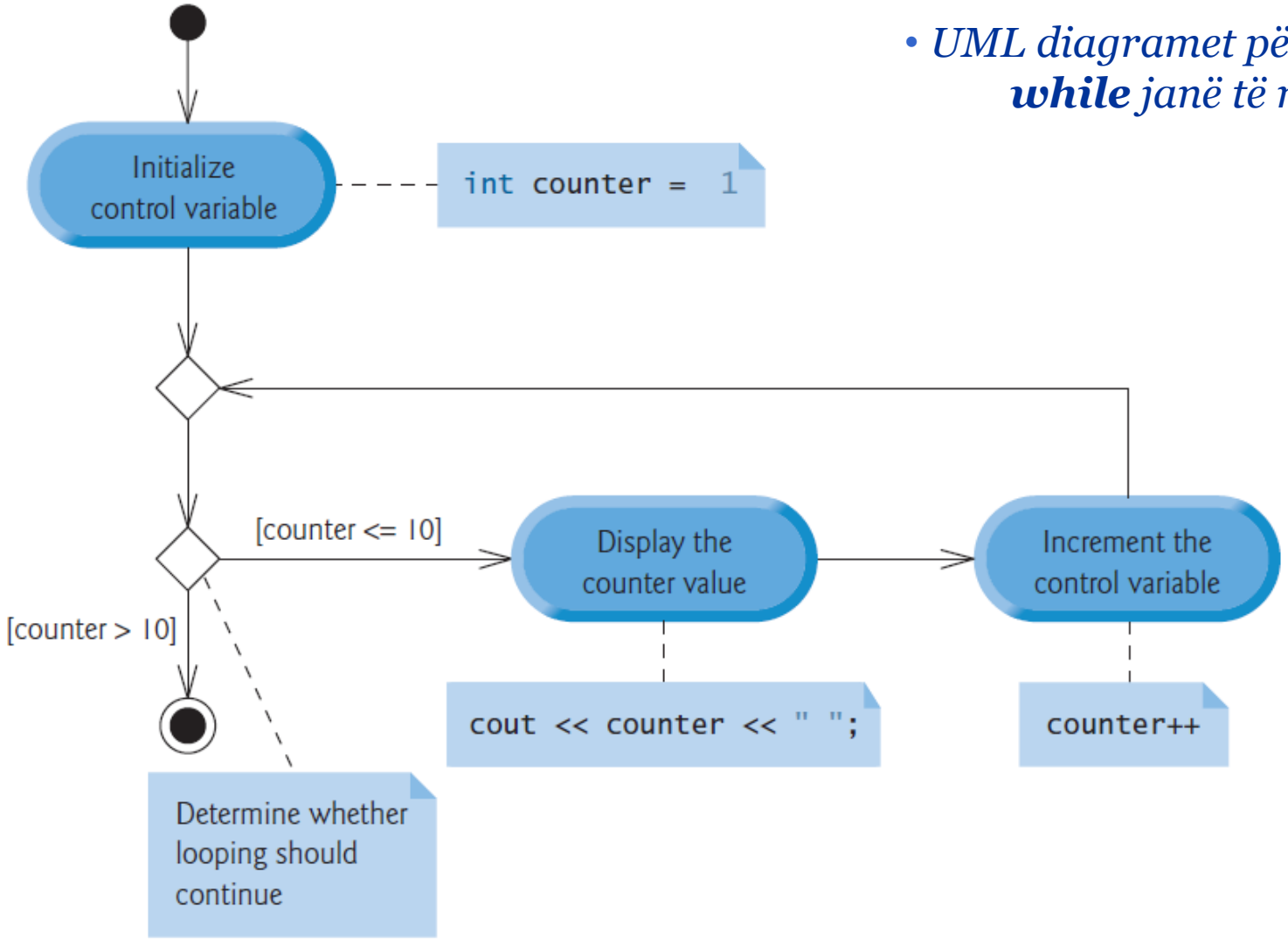
```
for ( initialization; loopContinuationCondition; increment )
    statement
```

or

```
initialization;

while ( loopContinuationCondition )
{
    statement
    increment;
}
```

for iteracioni

- **for** UML diagrami



- *UML diagramet për **for** dhe **while** janë të ngjajshme*

for iteracioni

- Shembuj:

```
for ( int i = 1; i <= 100; ++i )
```

```
for ( int i = 100; i >= 1; --i )
```

```
for ( int i = 7; i <= 77; i += 7 )
```

```
for ( int i = 20; i >= 2; i -= 2 )
```

for iteracioni

- Shembull: Mbledhja e numrave intexher nga 2 deri 20

```
#include <iostream>
using namespace std;

int main()
{
   int total = 0; // initialize total

   // total even integers from 2 through 20
   for ( int number = 2; number <= 20; number += 2 )
      total += number;

   cout << "Sum is " << total << endl; // display results
} // end main
```

```
Sum is 110
```

do ... while iteracioni

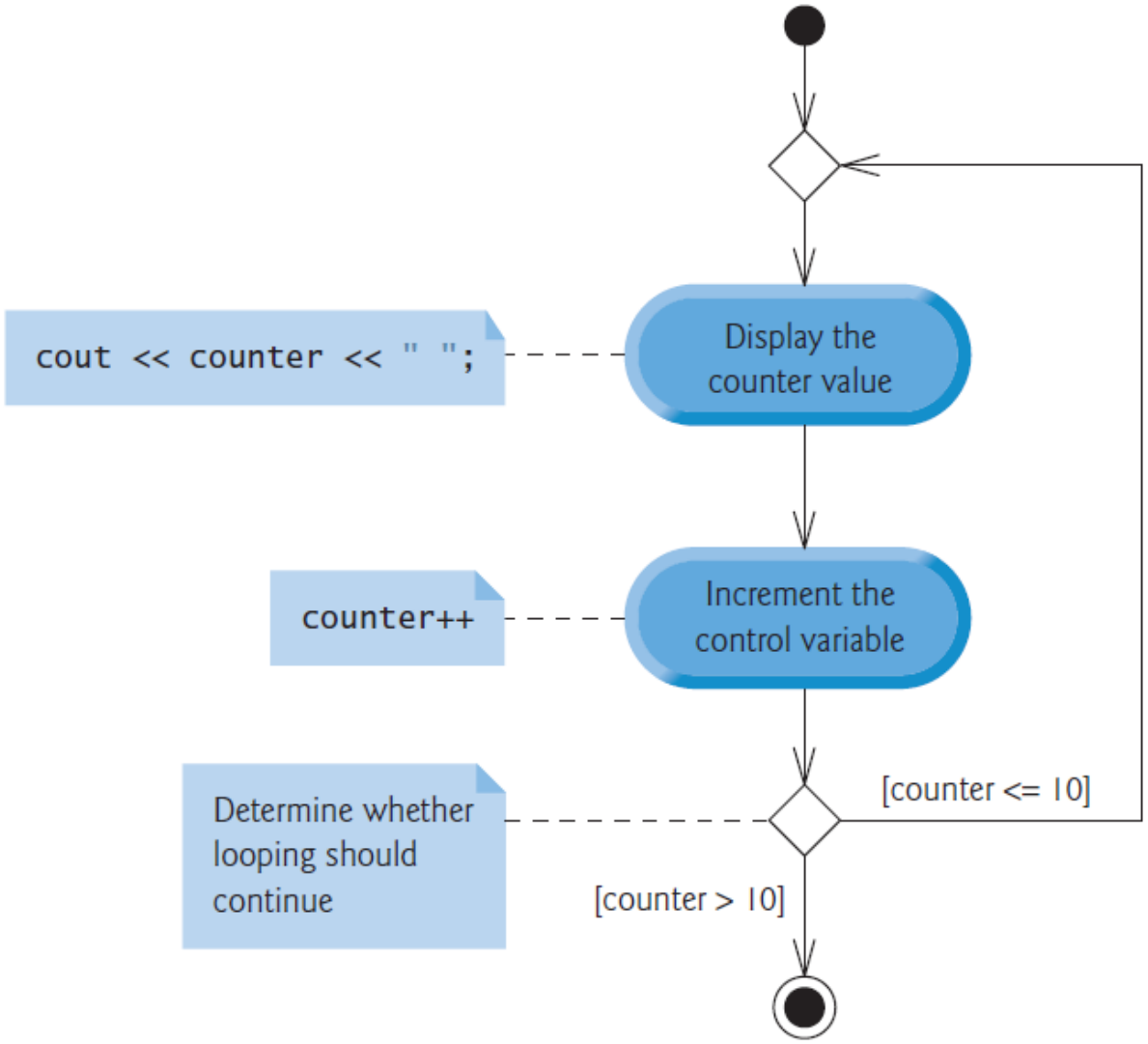
- **do...while** with one statement often is written as follows:

```
do
{
   statement
} while ( condition );
```

  - *The loop-continuation condition is not evaluated until after the loop performs its body at least once*

do ... while iteracioni

- UML diagram për **do...while** iteracione



do ... while iteracioni

- Shembull:

```
// do...while repetition statement.
#include <iostream>
using namespace std;

int main()
{
   int counter = 1; // initialize counter

   do
   {
      cout << counter << " "; // display counter
      ++counter; // increment counter
   } while ( counter <= 10 ); // end do...while

   cout << endl; // output a newline
} // end main
```

```
1 2 3 4 5 6 7 8 9 10
```

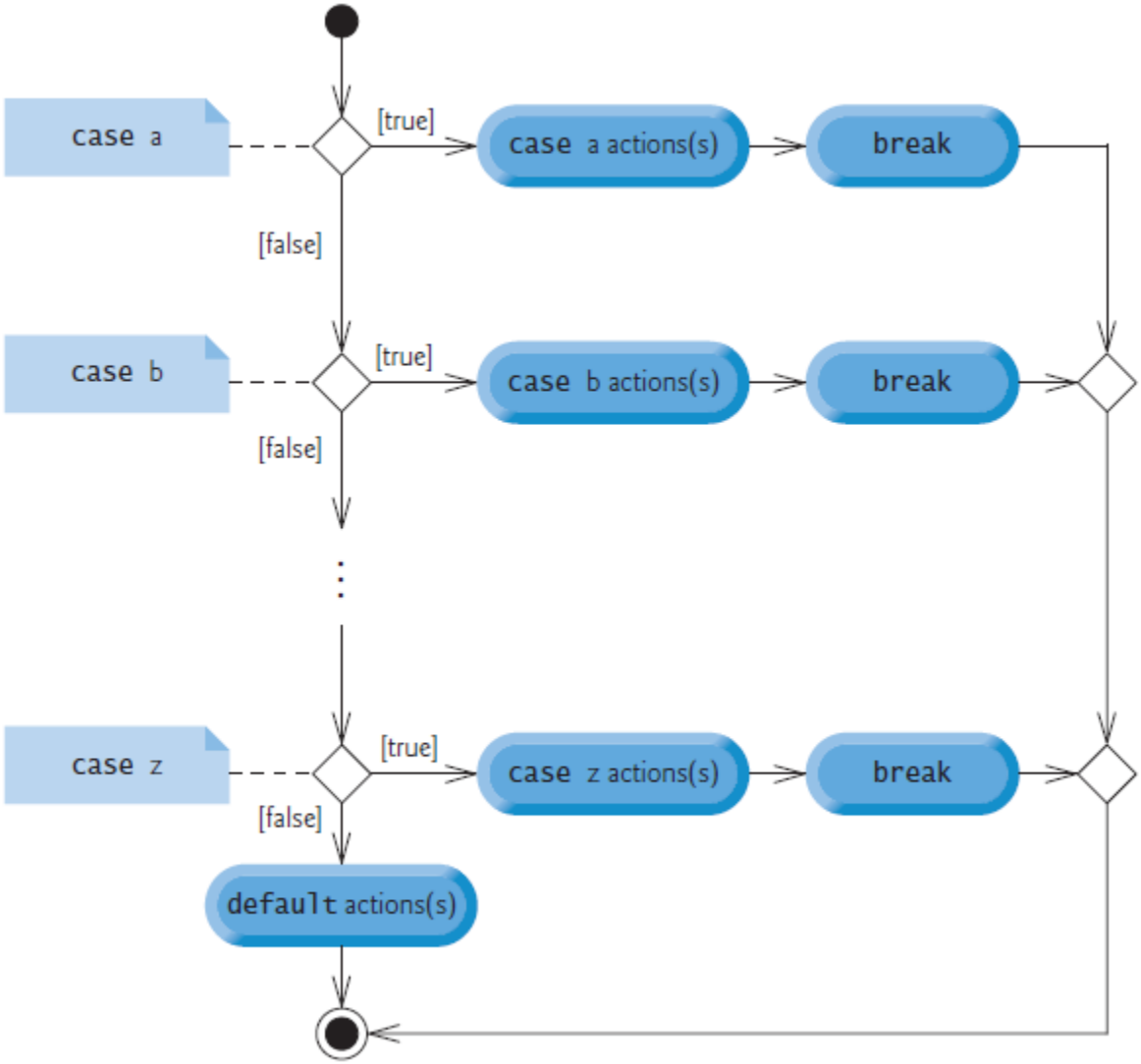
switch *zgjedhja e shumëfishtë*

- **switch** ofron zgjedhje të shumëfishtë varësisht nga vlerat në dispozicion të variablave ose të shprehjeve
- **switch** përmban një varg të kushteve të ashtuquajtura **case labels** dhe një opcion të përgjithshëm **default case** i cili vjen në shprehje nëse asnjëri nga kushtet paraprake nuk plotësohet

- Default Case
  - *Është opcionale dhe sështë patjetër të jetë në kuadër të* ***switch***

switch zgjedhja e shumëfishtë

- **switch** UML diagrami



break dhe continue

- **break**

- Ky urdhër, gjatë ekzekutimit të iteracioneve **while**, **for**, **do...while** ose **switch**, inicon braktisjen obligative (exit)

- Ekzekutimi i programit vazhdon me gjendjen e rradhës (next statement)

break dhe continue

- Shembull:

```
// break statement exiting a for statement.
#include <iostream>
using namespace std;

int main()
{
   int count; // control variable also used after loop terminates

   for ( count = 1; count <= 10; ++count ) // loop 10 times
   {
      if ( count == 5 )
         break; // break loop only if count is 5

      cout << count << " ";
   } // end for

   cout << "\nBroke out of loop at count = " << count << endl;
} // end main
```

```
1 2 3 4
Broke out of loop at count = 5
```

break dhe continue

- **continue**

- Ky urdhër, gjatë ekzekutimit të iteracioneve **while**, **for** ose **do...while**, anashkalon (skips) gjendjen aktuale dhe vazhdon ekzekutimin me gjendjen vijuese në kuadër të iteracionit (unazës) të njejtë (next iteration)

break dhe continue

- Shembull:

```
// continue statement terminating an iteration of a for statement
#include <iostream>
using namespace std;

int main()
{
   for ( int count = 1; count <= 10; ++count ) // loop 10 times
   {
      if ( count == 5 ) // if count is 5,
         continue;      // skip remaining code in loop

      cout << count << " ";
   } // end for

   cout << "\nUsed continue to skip printing 5" << endl;
} // end main
```

```
1 2 3 4 6 7 8 9 10
Used continue to skip printing 5
```

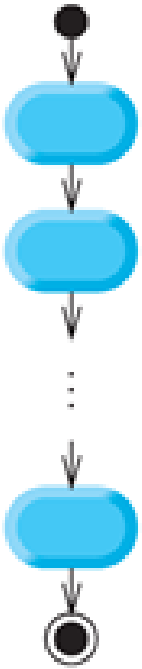
Struktura e kontrollit – Përfundim

- Programimi i strukturuar ofron mundësinë e të **kuptuarit** më të lehtë, të **testimit**, **debagimit**, **modifikimit** dhe **verifikimit** logjik/matematik të kodit të shkruar programor

- Në këtë rast, çdo rrjedhë e programit (*sekuencë*, *zgjedhje ose iteracion*) ka të vetmen pikë fillimi dhe të vetmen pikë përfundimi!

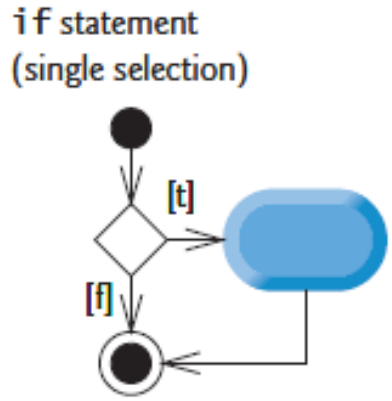
Struktura e kontrollit – Përfundim

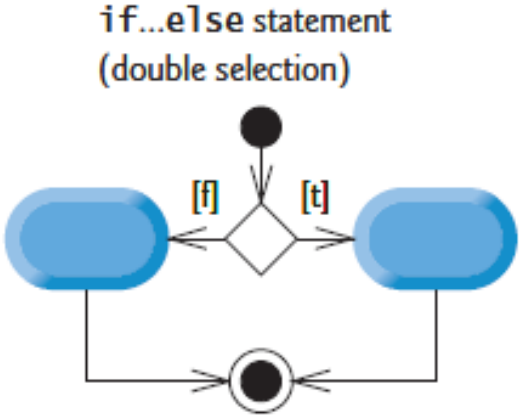
- Sekuencë:

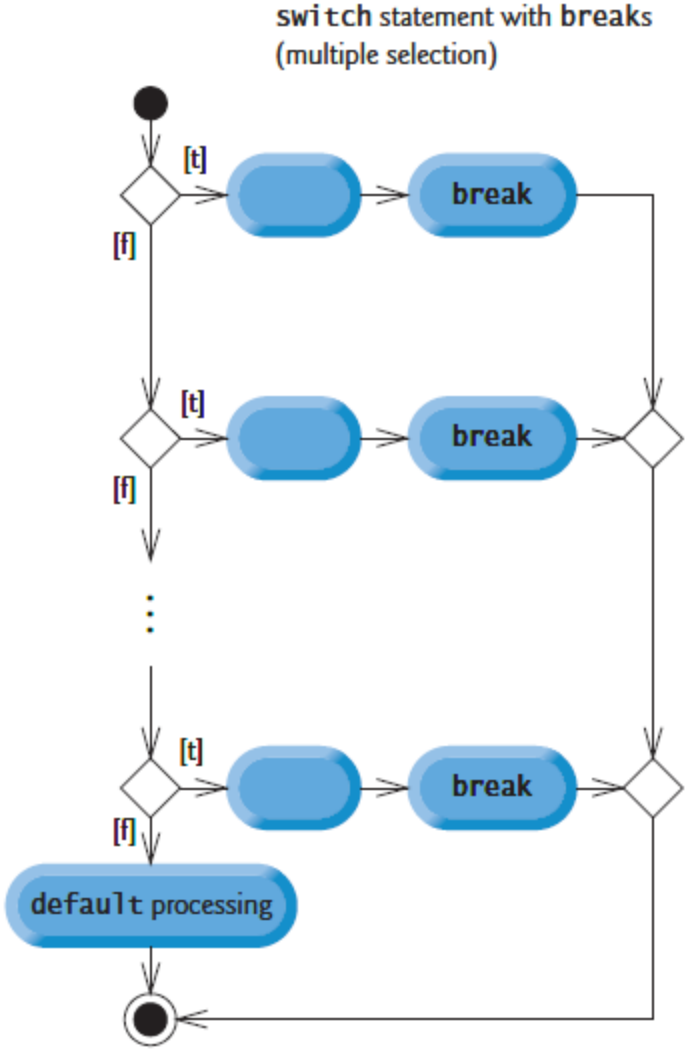


Struktura e kontrollit – Përfundim

- Selektim:

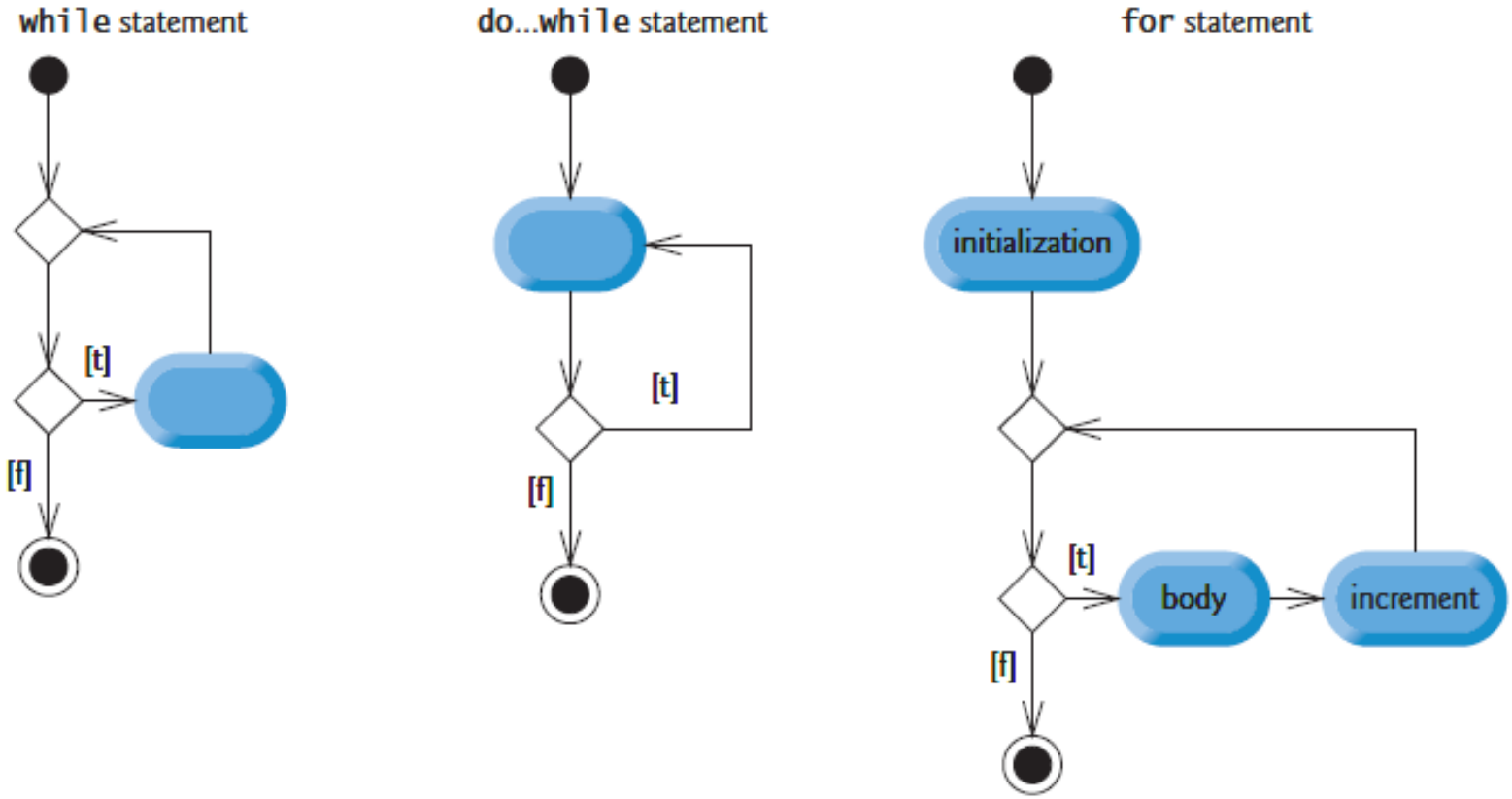






Struktura e kontrollit – Përfundim

- Iteracion:



- Pyetje eventuale?!

mentor.hamiti@universitetiaab.com