LECTURE 3:LOOP OPERATION

Dr. Alshimaa Hamdy

Assistant Professor Information Technology Dept., Faculty of Computer and Informatics, Tanta University

ORCID: orcid.org/0000-0003-4650-0484

Web of Science Researcher ID: publons.com/researcher/AAP-2239-2020/

Google Scholar Citation: Dr. Alshimaa Hamdy (Alshimaa H. Ismail)

Content

Repetition Statements

- Three types
  - `for` statement and `while` statement
    - Perform a group of actions zero or more times
  - `do..while` statement
    - Perform a group of actions at least once

`while` Repetition Statements

- Actions repeated while condition remains true
- Syntax

```
while (condition)
{
    action1;
    action2;
    .
    .
    actionN;
}
```

- One of the actions should causes condition to becomes false
- Example

```
int product = 3;

while ( product <= 30 )

      product *= 3;
```

Example

```
#include <stdio.h>

int main() {
  int i = 0;

  while (i < 5) {
    printf("%d\n", i);
    i++;
  }
    return 0;
}
```

`while` Repetition Statements (cont.)

- Not providing action that causes condition to becomes false
  - Lead to infinite loop
  - Logic error

- Example

```
int product = 3;
while ( product <= 30 )
      printf(product);
```

`for` Repetition Statement

- Provide counter-controlled repetition details in a single statement
- Syntax

```
for (initialization; loopContinuationCondition; update)
          action1;


for (initialization; loopContinuationCondition; update)
{
        action1; action2; … actionN;
}
```

- `for` loop repeats actions until condition becomes false

`for` Repetition Statement (cont.)

- When loop counter is declared in *initialization* expression, it can ONLY be used inside for statement (local variable)

- *initialization* and *update* expressions can be comma-separated lists of expressions

```
for(int i=0, j=0; i<4 && j<8; i++,j++)
    printf( "*" );
```

Examples Using `for` Statement

- Vary control variable from 100 to 1 in increments by -1 (decrement by 1)

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

- Vary control variable from 7 to 77 in steps of 7

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

- Vary control variable over the sequence: 99, 88, 77, 66, 55, 44, 33, 22, 11, 0

```
for(int i = 99; i >= 0; i -= 11 )
```

Example

```
#include <stdio.h>

int main() {
  int i;

  for (i = 0; i < 5; i++) {
    printf("%d\n", i);
  }

  return 0;
}
```

Examples Using `for` Statement (cont.)

- Using a comma-separated list of expressions

```
int total =0;

for ( int number = 2; // initialization

        number <= 20; // loop continuation condition

        total += number, number += 2 ) // total and increment

    ;  // empty statement
```

can be written as

```
int total =0;
for ( int number = 2; number <= 20; number += 2 )
     total += number;
```

`do..while` Repetition Statement

- Similar to `while` statement but `while` statement tests loop-continuation before performing body of loop
- `do..while` tests loop-continuation after performing body of loop
  - Loop body always executes at least once
- Syntax

```
do
{
    action1;
    action2;
    .
    .
    actionN;
} while (condition)
```

Example

```
#include <stdio.h>

int main() {
  int i = 0;

  do {
    printf("%d\n", i);
    i++;
  }
  while (i < 5);

  return 0;
}
```

`break` Statement

- ▶Alter flow of control
  - ▶Causes immediate exit from control structure

- Used with `while`, `for`, `do…while` or `switch` statements
  - ▶Escape early from a loop (`while, for, do…while`)
  - ▶Skip the remainder of `switch`

Example

```
#include <stdio.h>

int main() {
  int i;

  for (i = 0; i < 10; i++) {
    if (i == 4) {
      break;
    }
    printf("%d\n", i);
  }

  return 0;
}
```

`continue` Statement

- Used with `while`, `for` or `do…while` statements

▶Alter flow of control

  - ▶Skips remainder of loop body of current iteration
  - ▶Proceeds with next iteration of loop

- With `while` and `do…while` statements
  - ▶Loop-continuation test is evaluated immediately after continue statement

- With `for` statement
  - ▶*Update* expression is executed
  - ▶Next, loop-continuation test is evaluated

Example

```
#include <stdio.h>

int main() {
  int i;

  for (i = 0; i < 10; i++) {
    if (i == 4) {
      continue;
    }
    printf("%d\n", i);
  }

  return 0;
}
```

Thank you