Lab 4 - Felsökning - Tobias Skog .NET 23

Tobias Skog .NET 23

Case 1

Problem

> Koden genererar ett fel och körs inte.

Problem Kod

```
1  int number = 2;
2
3  if (number > 3)
4  {
5      Console.WriteLine("Talet är större än tre")
6  }
7  elseif(number < 3)
8  {
9      Console.WriteLine("Talet är mindre än tre");
10 }
```

Felbeksrivning

På rad 5 är det ett uteslutet `;` .
På rad 7 så är problemet `elseif` . I C# (och många andra språk) använder vi `else if` .
En fördel med detta är att koden blir lite lättare att läsa.

> Om vi jämför `if ... else if ... else` med `if ... elseif ... else`
> Så är det lättare att se strukturen i det första exemplet.

Uppdaterad Kod

```
1  int number = 2;
2
3  if (number > 3)
4  {
5      Console.WriteLine("Talet är större än tre");
6  }
7  else if(number < 3)
8  {
9      Console.WriteLine("Talet är mindre än tre");
10 }
```

Tobias Skog .NET 23

Case 2

Problem

> Koden ska skriva ut alla siffror mellan 1 och 100 men skriver bara ut till nummer 99

Problem Kod

```
1  for(int i=1; i<100; i++)
2  {
3      Console.WriteLine(i);
4  }
```

Felbeksrivning

På rad 1 i våran for loop så börjar vi korrekt med att räkna från 1 upp till `< 100` = 99.
Det vi måste tänka på då vi vill skriva ut den sista siffran i vår for loop är att ändra `<` till `⩽` då ser for loopen ut så här `for(int i = 1; i ⩽ 100; i++)` Då börjar vi på 1 och fortsätter tills dess att `i` är mindre ELLER likamed 100. dvs sista iterationen av loopen så är `i = 100`, och vi skriver `100` till konsolen.

Uppdaterad Kod

```
1  for(int i=1; i⩽100; i++)
2  {
3      Console.WriteLine(i);
4  }
```

Tobias Skog .NET 23

Case 3

Problem

> Koden genererar ett fel och körs inte

Problem Kod

```
1  for (int i = 1; i ⩽ 5; i)
2  {
3      for (int j = 1; j ⩽ i; j)
4      {
5          Console.Write(j + " ");
6      }
7      Console.WriteLine();
8  }
```

Felbeksrivning

På rad 1 i slutet av for loopen har vi glömt att välja vad vi ska göra med vår variable `i`.
På rad 3 har vi samma problem fast med variabeln `j`.
Då vi initiar `i` och `j` till `1` så antar jag att vi vill inkrementera loopen med `1` efter varje iteration.
Så lösningen blir:
Rad 1 `for (int i = 1; i ⩽ 5; i++)`
Rad 3 `for (int j = 1; j ⩽ i; j++)`
Notera att vi la till `++` i slutet av for loopen till vår variabel `i` och `j`.

Uppdaterad Kod

```
1  for (int i = 1; i ⩽ 5; i++)
2  {
3      for (int j = 1; j ⩽ i; j++)
4      {
5          Console.Write(j + " ");
6      }
7      Console.WriteLine();
8  }
```

Tobias Skog .NET 23

Case 4

Problem

> Koden har ingen output, varför inte?

Problem Kod

```
1  int i = 1;
2  while (i <= 5)
3  {
4          for (int j = 1; j <= i; j++)
5          {
6                  Console.Write();
7          }
8
9          Console.WriteLine();
10         i++;
11 }
```

Felbeksrivning

På rad 6 använder vi oss av `Console.Write()` , men vi ger inte metoden någon data att skriva.
`Console.Write()` metoden har ingen overload metod där den tar 0 argument, så glömmer vi att ge metoden någon form av data att skriva ut så blir det ett error.
Om vi istället passar in variabeln `j` så får vi följande output.

```
1  1
2  12
3  123
4  1234
5  12345
```

eller om vi följer koden i Case 3 och skriver ut `Console.Write(j + " ");` så får vi följande output

```
1  1
2  1 2
3  1 2 3
4  1 2 3 4
5  1 2 3 4 5
```

Det andra exemplet ser visuelt bättre ut, så det är lösningen jag kommer att ha i min uppdaterade kod

Uppdaterad Kod

```
1  int i = 1;
2  while (i <= 5)
3  {
4          for (int j = 1; j <= i; j++)
5          {
6                  Console.Write(j + " ");
7          }
8
9          Console.WriteLine();
10         i++;
11 }
```

Tobias Skog .NET 23

Case 5

Problem

> Eftersom i är 1 i exemplet nedan förväntar vi oss att koden inte ska göra något men nu skriver den ut "Two". Varför blir det så och hur löser vi det så att det inte blir någon utskrift om i är 1?

Problem Kod

```
1  int i = 1;
2
3  switch (i)
4  {
5      case 1:
6      case 2:
7          Console.WriteLine("Two");
8          break;
9      default:
10         Console.WriteLine("Other");
11         break;
12 }
```

Felbeksrivning

När man använder sig av en `switch` kan man sätta flera alternativ till samma svar genom att sätta flera `case` ´s efter varandra som i koden här ovan så chainar vi `case 1:` och `case 2:` tillsammans och säger OM `i` är `1` ELLER `2` så gör detta sen `default:` är vad som händer om `i` är allt annat än `1` eller `2` .
I den uppdaterade koden så har vi lagt till 1 rad kod (rad 6). Efter `case 1:` som också kan läsas som (`i = 1`) så avbryter vi switchen med `break` och skriver inte ut något till konsolen.

Uppdaterad Kod

```
1  int i = 1;
2
3  switch (i)
4  {
5      case 1:
6              break;
7      case 2:
8          Console.WriteLine("Two");
9          break;
10     default:
11         Console.WriteLine("Other");
12         break;
13 }
```

Tobias Skog .NET 23

Case 6

Problem

> Koden genererar ett fel och körs inte.

Problem Kod

```
int i = 10;

if(i = 5)
{
    Console.WriteLine("i är 5");
}
```

Felbeksrivning

På raid 3 så uttrycker vi oss fel i if satsen.
När vi vill kolla om ett värde är desamma som ett annat använder vi oss av `==`.
När vi tilldelar ett värde till en variabel använder vi oss av `=`.

Uppdaterad Kod

```
int i = 10;

if(i == 5)
{
    Console.WriteLine("i är 5");
}
```