TDP004 - Objektorienterad programmering

Styrstrukturer, funktioner och strömmar

Pontus Haglund & Rasmus Jonsson

Institutionen för datavetenskap

LINKÖPING UNIVERSITY

1. Mål med föreläsningen
2. Logiska operatorer & styrstrukturer
3. Funktioner och scope
4. Mer const
5. Mer om buffrar
6. Cppreference
7. Referens

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

Mål med föreläsningen

Efter föreläsningen skall studenten kunna:

- Hantera logiska operatorer.
- Bygga styrstrukturer.
- Hantera funktioner och scope.
- Använda const vid rätt tillfälle.
- Skriva till `std::cerr`.
- Hänvisa rättningsprotokoll.

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

Logiska operatorer & styrstrukturer

Logiska operatorer

Python

- `not a`
- `a == b`
- `a != b`
- `a < b`
- `a <= b`
- `a > b`
- `a >= b`

Logiska operatorer & styrstrukturer

Logiska operatorer

Python

- `not a`
- `a == b`
- `a != b`
- `a < b`
- `a <= b`
- `a > b`
- `a >= b`

C++

- `!a`
- `a == b`
- `a != b`
- `a < b`
- `a <= b`
- `a > b`
- `a >= b`

Logiska operatorer & styrstrukturer

Kombinerade logiska operatorer

Python

- `a <= b and b >= c`
- `a == b or b == c`

Logiska operatorer & styrstrukturer

Kombinerade logiska operatorer

Python

- `a <= b and b >= c`
- `a == b or b == c`

C++

- `a <= b && b >= c`
- `a == b || b == c`

Grundläggande styrstrukturer

Styrstrukturer

Python

```
if 1 != 1:
    print(1)
elif True:
    print(2)
else:
    print(3)
```

Grundläggande styrstrukturer

Styrstrukturer

Python

```python
if 1 != 1:
    print(1)
elif True:
    print(2)
else:
    print(3)
```

C++

```cpp
if (1 != 1)
{
  cout << 1 << endl;
}
else if (true)
{
  cout << 2 << endl;
}
else
{
  cout << 3 << endl;
}
```

Logiska operatorer & styrstrukturer

Loopar

Python

- `while`
- `for`
-

Logiska operatorer & styrstrukturer

Loopar

Python

- `while`
- `for`
-

C++

- `while`
- `for`
- `do-while`

Grundläggande styrstrukturer

while-loop

Python

```
i = 0

while i <= 2:
    print(i)
    i += 1
```

Grundläggande styrstrukturer

while-loop

Python

```
i = 0

while i <= 2:
    print(i)
    i += 1
```

C++

```
int i{0};

while(i <= 2)
{
  cout << i << endl;
  i++;
}
```

Grundläggande styrstrukturer

for-loop

Python

```
for i in range(0,2):
    print(i)
```

Grundläggande styrstrukturer

for-loop

Python

```
for i in range(0,2):
    print(i)
```

C++

```
for(int i{0}; i < 2; ++i)
{
  cout << i << endl;
}
```

Logiska operatorer & styrstrukturer

do-while-loop

```
int i{0};

do
{
  cin >> i;
}
while(i != 1);
```

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

Funktioner och scope

Block

```
{ // Början av blocket

  // Blockets kroppen

} // Slutet av blocket
```

Funktioner och scope

Scope

```
int x{}; // Globalt scope
int main()
{
  int y{}; // Lokalt scope
  return 0;
}
```

Funktioner och scope

Block och scope

```
int x{0};
{
  int x{1};
  {
    cout << x << ' ';
    int x{2};
    cout << x << ' ';
  }
  cout << x << ' ';
}
cout << x << ' '
     << endl;
```

Funktioner och scope

Block och scope

```
int x{0};
{
  int x{1};
  {
    cout << x << ' ';
    int x{2};
    cout << x << ' ';
  }
  cout << x << ' ';
}
cout << x << ' '
     << endl;
```

```
$ ./a.out
1 2 1 0
```

Funktioner och scope

Mer block och scope

```
int x{0};

int main()
{
  int y{1};
  {
    int z{2};
    cout << x << ' ' << y << ' ' << z << endl;
  }
}
```

Funktioner och scope

Funktioner

```
def add(left, right):
    return left + right
```

Funktioner och scope

Funktioner

```
def add(left, right):
    return left + right
```

```
int add(int left, int right)
{
  return left + right;
}
```

Funktioner och scope

Funktions skelett

```
retur_typ funktions_namn(parametrar)
{
  //uttryck
  return resultat;
}
```

Funktioner och scope

Procedurer (funktioner utan retur)

```
void foo()
{
  cout << "En procedur" << endl;
}
```

Funktioner och scope

Deklaration och definition

```
void hello(); // deklaration


// ...
// ...


void hello() // definition
{
  cout << "hello" << endl;
}
```

Funktioner och scope

Deklaration och definition

```
void hello(); // deklaration

int main()
{
  hello();
  return 0;
}

void hello() // definition
{
  cout << "hello" << endl;
}
```

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

Mer const

Parameteröverföring

```
void hello(string name)
{
  cout << "Hello " << name
       << endl;
}

int main()
{
  string name{"Pontus"};
  hello(name);
  return 0;
}
```

Mer const

Parameteröverföring

```
void hello(string name)
{
  cout << "Hello " << name
       << endl;
}

int main()
{
  string name{"Pontus"};
  hello(name);
  return 0;
}
```

```
$ ./a.out
Hello Pontus
```

Mer const

Parameteröverföring

```
void hello(string const name)
{
  cout << "Hello " << name
       << endl;
}

int main()
{
  string name{"Pontus"};
  hello(name);
  return 0;
}
```

```
$ ./a.out
Hello Pontus
```

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

Mer om buffrar

cerr

```
int main()
{
  cerr << "Something went wrong" << endl;
  cout << "Something went right" << endl;
  return 0;
}
```

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

Cppreference

Cppreference

Mer information om standard biblioteket kan ni hitta på:
https://en.cppreference.com/w/

1 Mål med föreläsningen
2 Logiska operatorer & styrstrukturer
3 Funktioner och scope
4 Mer const
5 Mer om buffrar
6 Cppreference
7 Referens

LINKÖPING UNIVERSITY

Referens

Referens vid parameteröverföring

```
void hello(string name)
{
  cout << "Hello " << name
       << endl;
}

int main()
{
  string name{"Pontus"};
  hello(name);
  return 0;
}
```

Referens

Referens vid parameteröverföring

```
void hello(string & name)
{
  cout << "Hello " << name
       << endl;
}

int main()
{
  string name{"Pontus"};
  hello(name);
  return 0;
}
```

Referens

Referens vid parameteröverföring

```
void hello(string const& name)
{
  cout << "Hello " << name
       << endl;
}

int main()
{
  string name{"Pontus"};
  hello(name);
  return 0;
}
```

www.liu.se

LINKÖPING UNIVERSITY