Programmierparadigmen

PropaScript

```
a = 5
```

Variablen & Zuweisung

```
a = b = 8
```

Kettenzuweisung

```
i = 1
while (i - 10) {
  print i * i * i
  i = i - (0 - 1)
}
```

Turingvollständig :D

```
print 1
print 2
```

Statements & Ausgabe

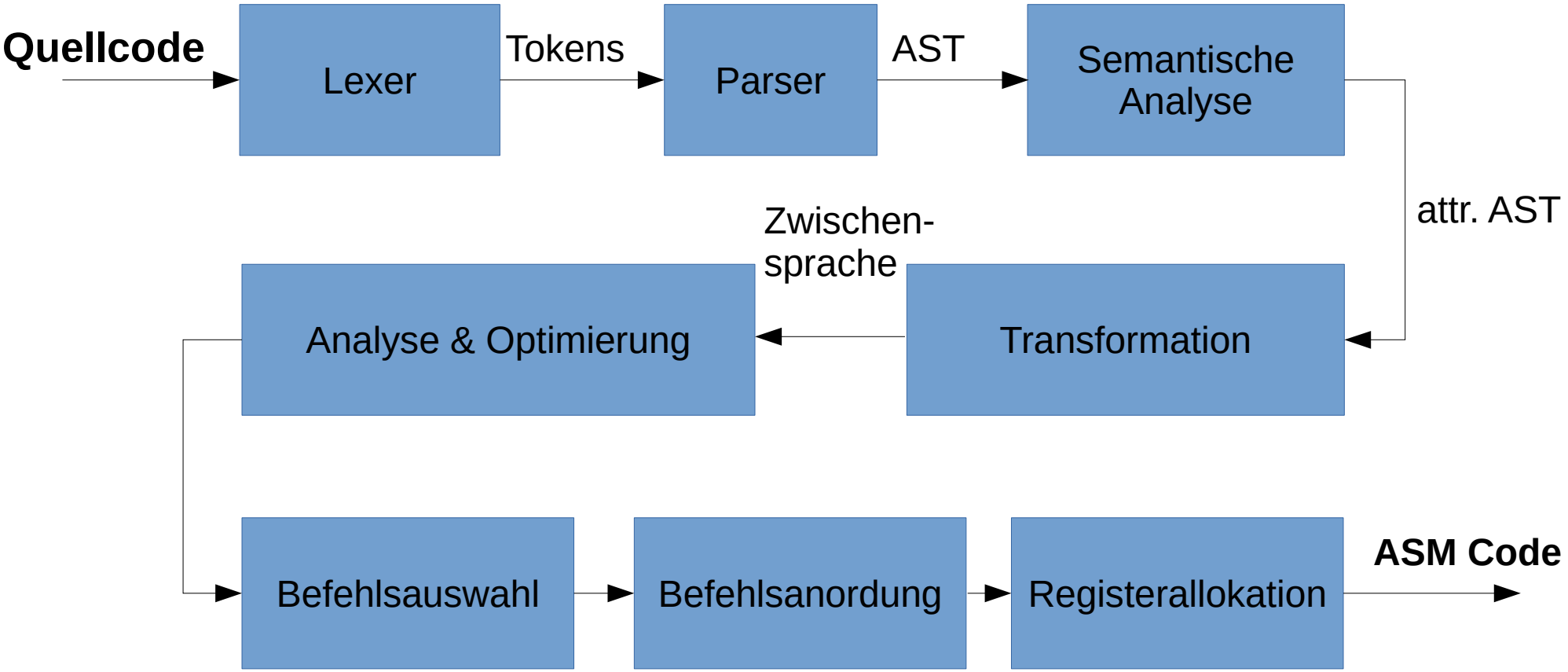
```
while (1) {
    print 2
}
```

Schleifen

```
print b * (7 - 2)
```

Multiplikation &
Subtraktion

Compiler Pipeline



Straight aus der Compilerbau Vorlesung geklaut

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program → Statements eof

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program → Statements eof

Statements → Statement eos Statements | ε

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program → Statements eof

Statements → Statement eos Statements | ε

Statement → while Expr { Statements }
| Expr
| print Expr

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	Value | Value Op Expr
Op	→	= | * | -

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	Value | Value Op Expr
Op	→	= | * | -
Value	→	(Expr) | Identifier | Number

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program → Statements eof

Statements → Statement eos Statements | ε

Statement → while Expr { Statements }
| Expr
| print Expr

Expr → AssignExpr

Value → (Expr) | Identifier | Number

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	AssignExpr
AssignExpr	→	MinusExpr = Expr | MinusExpr
Value	→	(Expr) | Identifier | Number

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	AssignExpr
AssignExpr	→	MinusExpr = Expr | MinusExpr
MinusExpr	→	MulExpr - MinusExpr | MulExpr
Value	→	(Expr) | Identifier | Number

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	AssignExpr
AssignExpr	→	MinusExpr = Expr | MinusExpr
MinusExpr	→	MulExpr - MinusExpr | MulExpr
MulExpr	→	Value * MulExpr | Value
Value	→	(Expr) | Identifier | Number

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	AssignExpr
AssignExpr	→	MinusExpr = Expr | MinusExpr
MinusExpr	→	MulExpr - MinusExpr | MulExpr
MulExpr	→	Value * MulExpr | Value
Value	→	(Expr) | Identifier | Number

Jedes grammatikalisch korrekte Programm ist auch semantisch korrekt?

Grammatik

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	AssignExpr
AssignExpr	→	MinusExpr = Expr | MinusExpr
MinusExpr	→	MulExpr - MinusExpr | MulExpr
MulExpr	→	Value * MulExpr | Value
Value	→	(Expr) | Identifier | Number

Grammatik erlaubt:
5 * 4 = 8 * 9

First & Follow Mengen

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Program	→	Statements eof
Statements	→	Statement eos Statements | ε
Statement	→	while Expr { Statements } | Expr | print Expr
Expr	→	AssignExpr
AssignExpr	→	MinusExpr = Expr | MinusExpr
MinusExpr	→	MulExpr - MinusExpr | MulExpr
MulExpr	→	Value * MulExpr | Value
Value	→	(Expr) | Identifier | Number

Linksfaktorisierung

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Abstiegsparser kann nicht entscheiden in welche Alternative er absteigt.

Expr → AssignExpr

AssignExpr → MinusExpr = Expr | MinusExpr

Linksfaktorisierung

while	print	(	)	{	}	=	*	-	Number	Identifier	eos	eof

Expr → AssignExpr

AssignExpr → MinusExpr AssignExpr’

AssignExpr’ → = Expr | ε

Ast

- Wie das Programm intern repräsentieren?

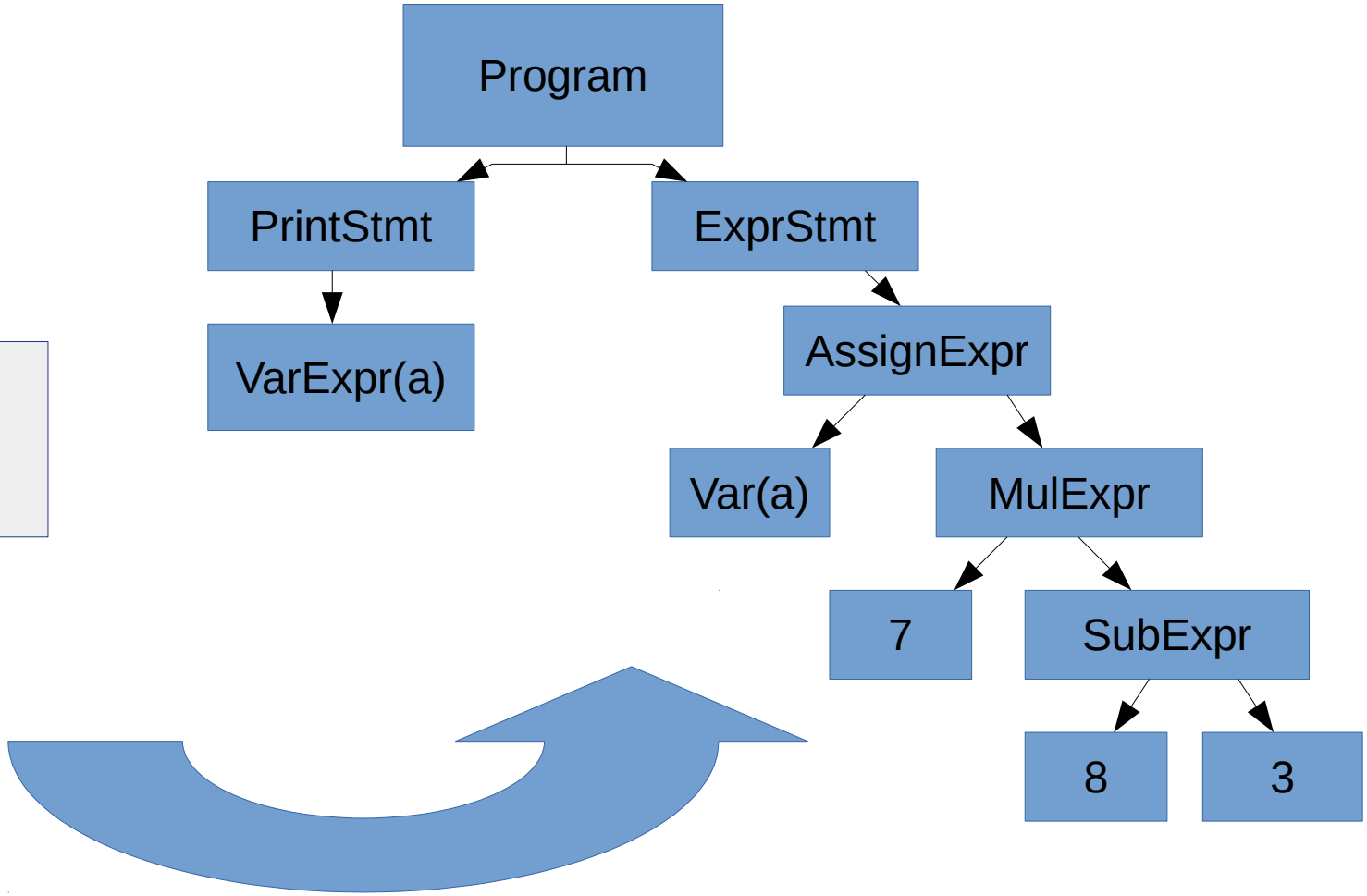
```
a = 7 * (8 - 3)
print a
```

?

Ast

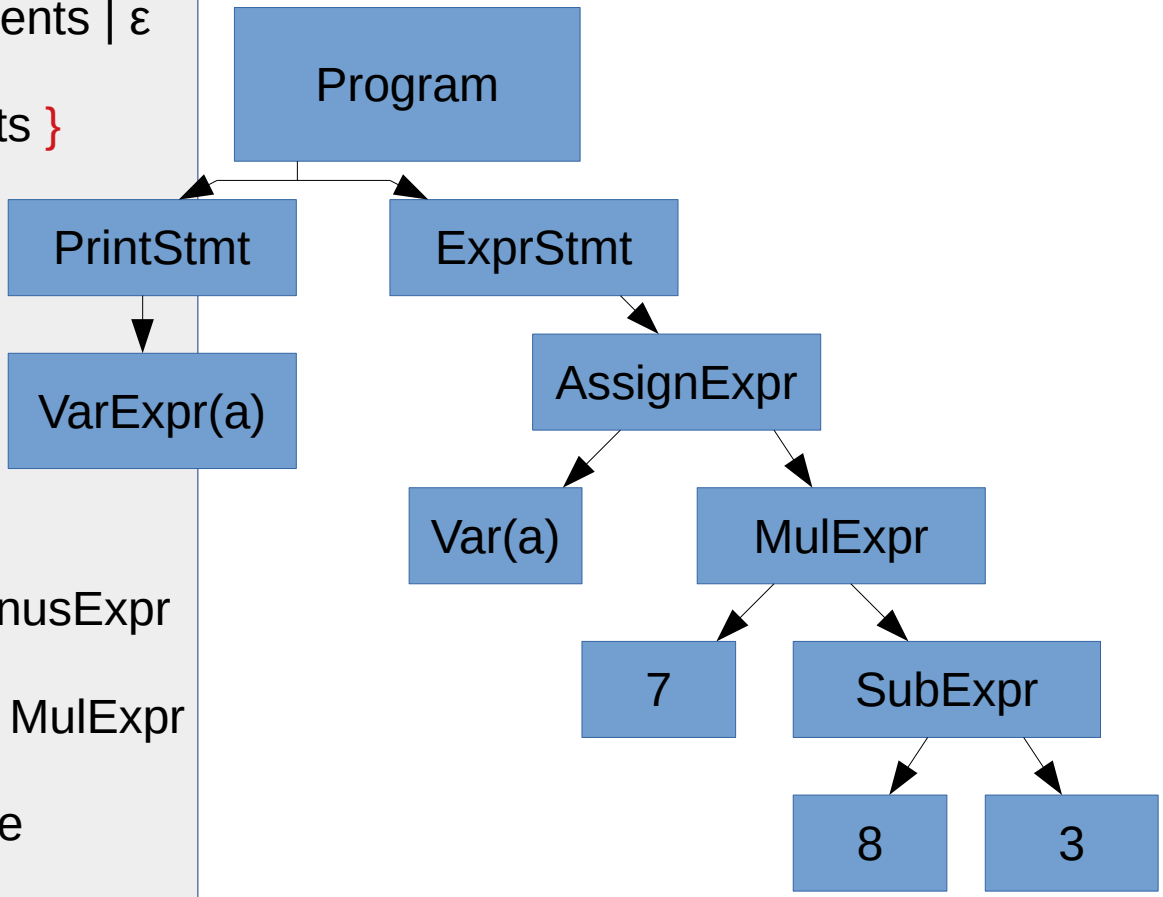
- Repräsentiere Program als Abstrakten Syntax Baum

```
a = 7 * (8 - 3)
print a
```

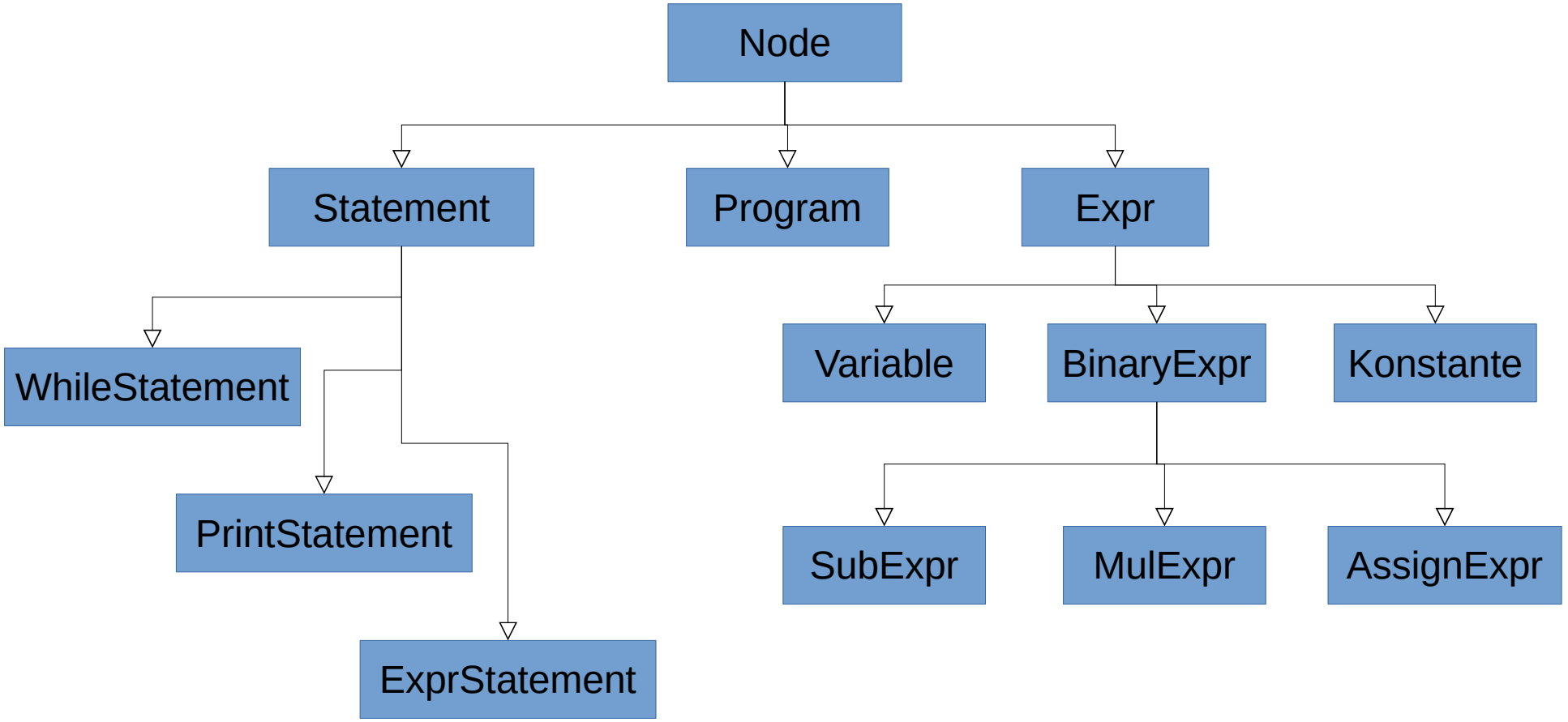


Ast

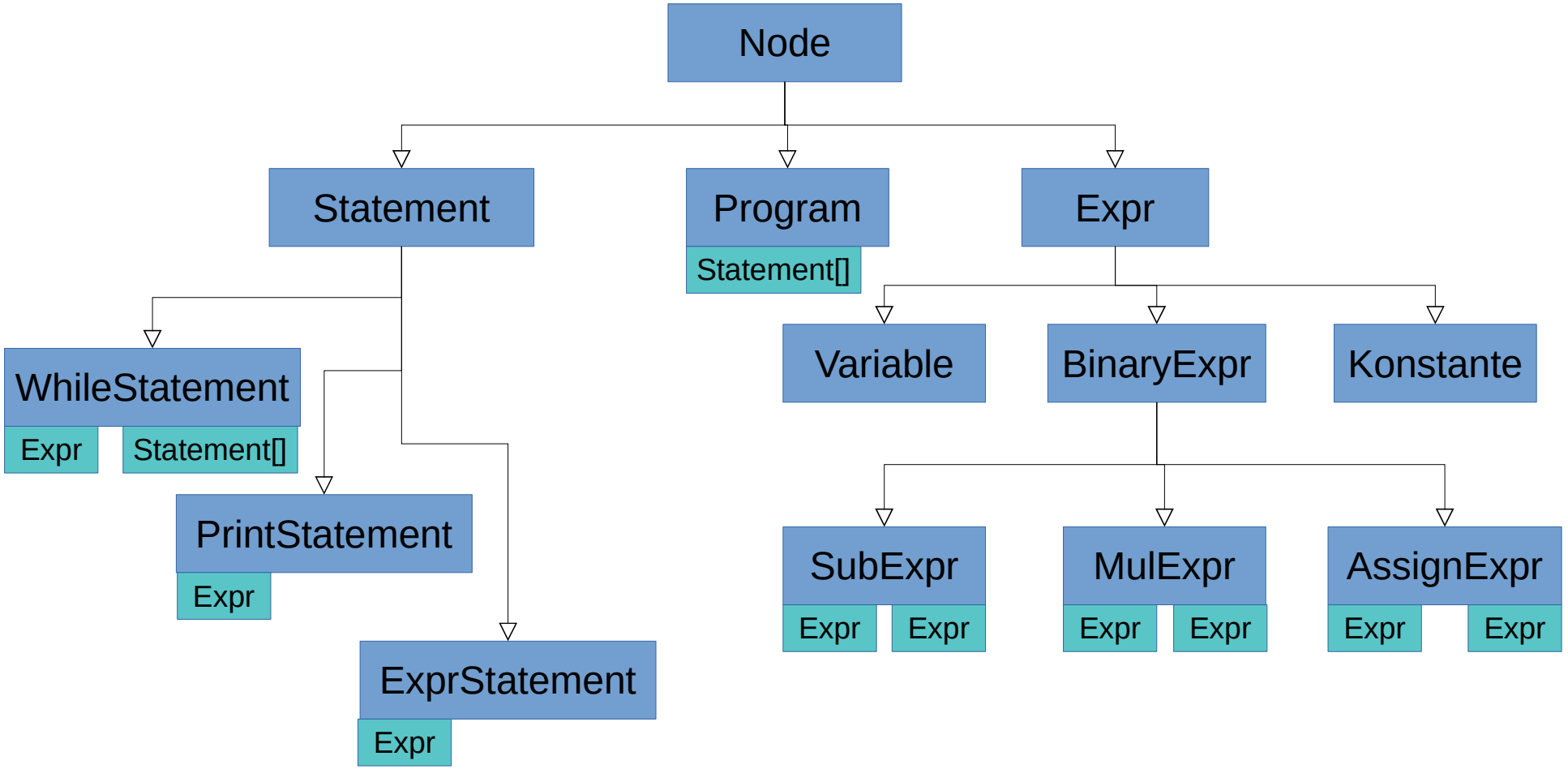
```
Program      →  Statements eof

Statements   →  Statement eos Statements | ε

Statement    →  while Expr { Statements }
                |  Expr
                |  print Expr

Expr         →  AssignExpr

AssignExpr   →  MinusExpr = Expr | MinusExpr

MinusExpr    →  MulExpr - MinusExpr | MulExpr

MulExpr      →  Value * MulExpr | Value

Value        →  ( Expr ) | Identifier | Number
```



Ast VererbungsHierarchie



Ast VererbungsHierarchie



Java Bytecode

```
a = 7 * (8 - 3)
print a
```

?

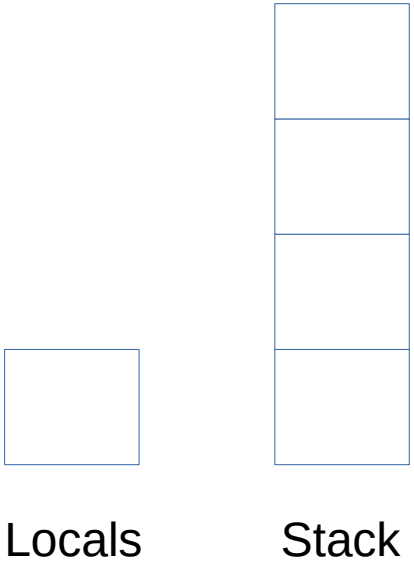
Java Bytecode

```
a = 7 * (8 - 3)
print a
```

```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

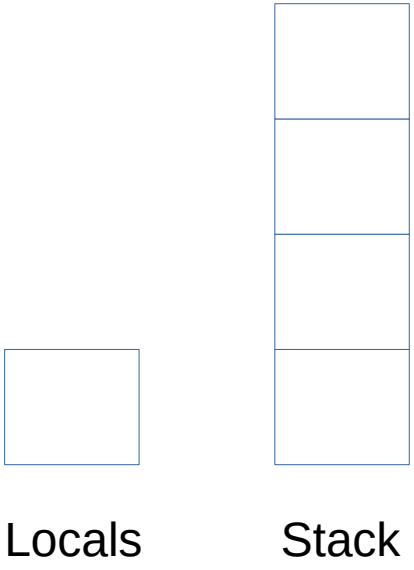
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

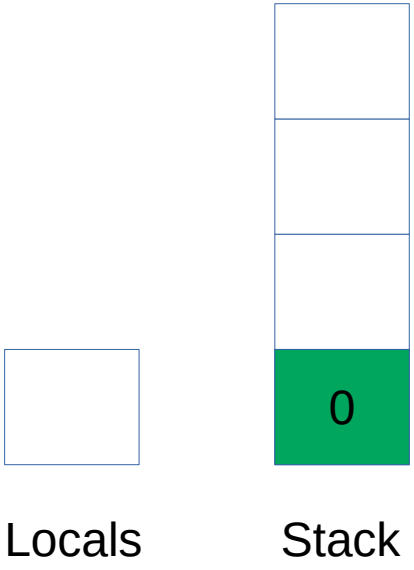
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

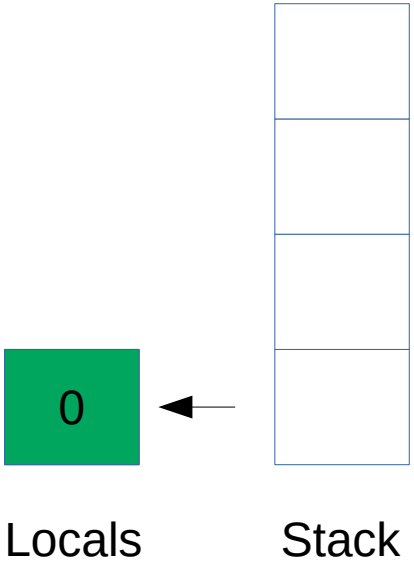
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

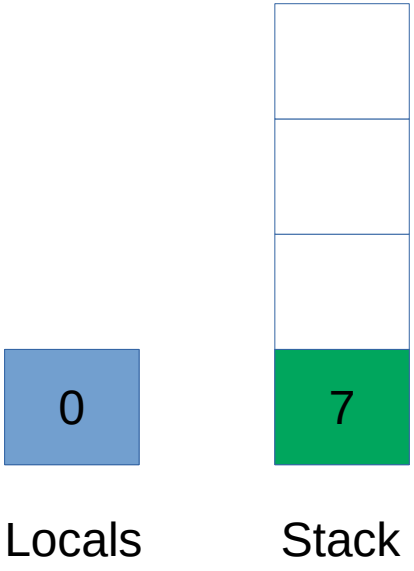
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

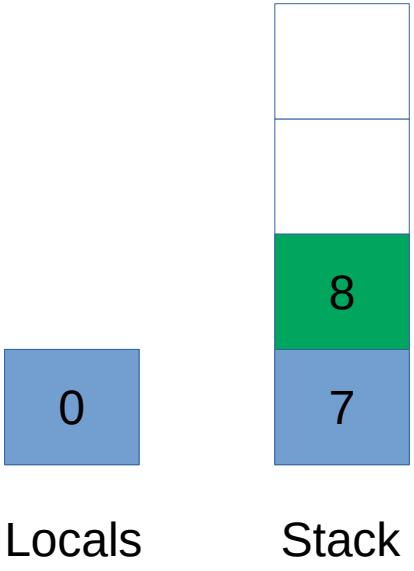
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

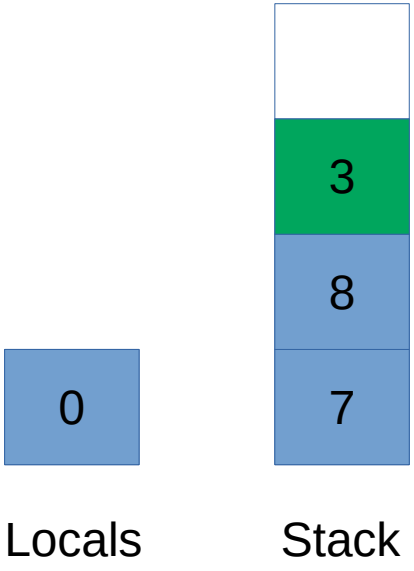
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

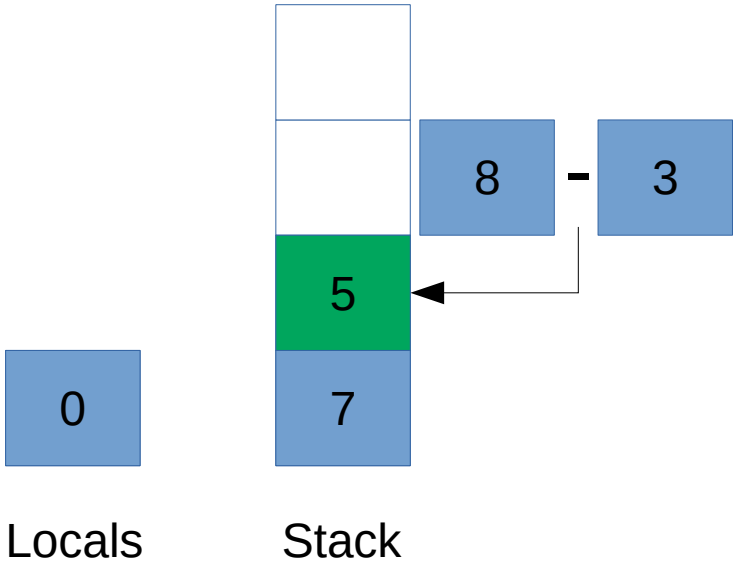
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

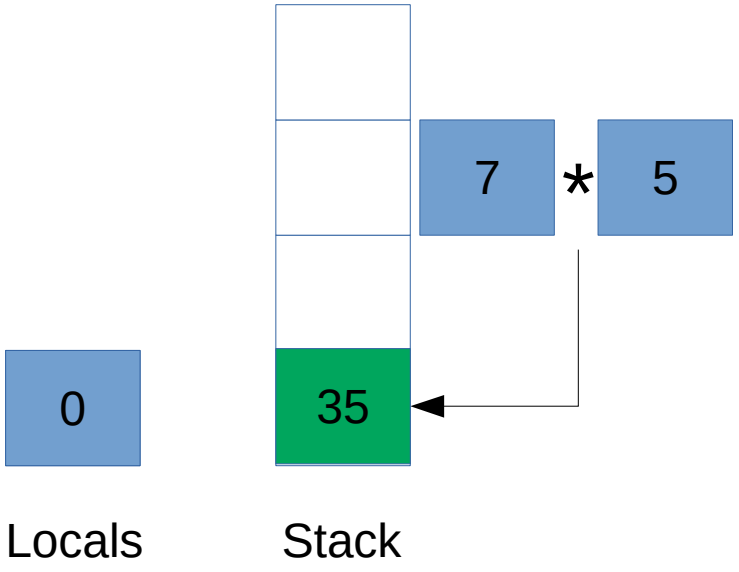
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

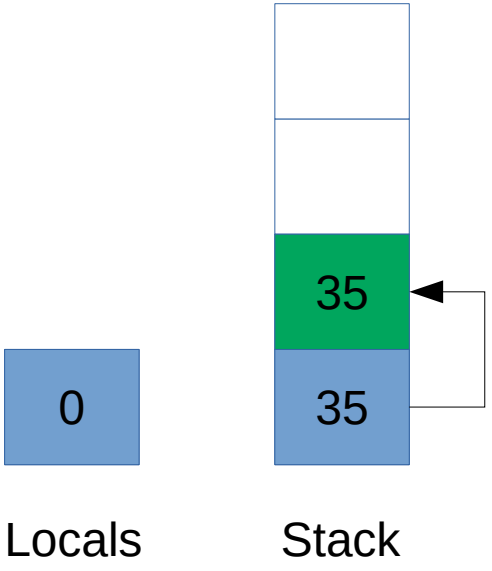
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

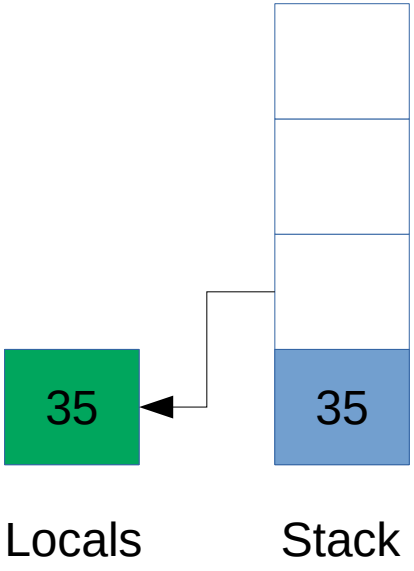
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

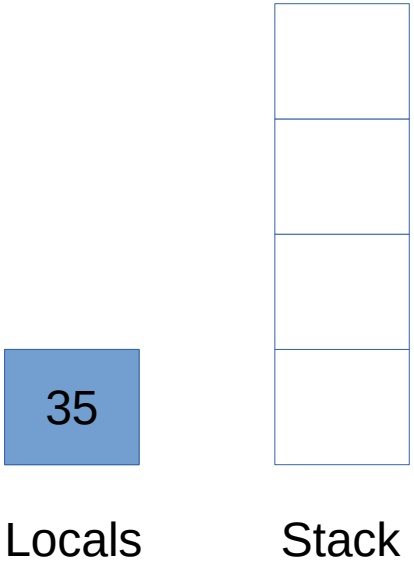
```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Java Bytecode

```
a = 7 * (8 - 3)
print a
```




```
.method public static main([Ljava/lang/String;)V
    .limit stack 4
    .limit locals 1

    ; Initialize local variables
    iconst_0
    istore 0

    ; a = 7 * (8 - 3);
    ldc 7
    ldc 8
    ldc 3
    isub        ; 8 - 3
    imul        ; 7 * (8 - 3)
    dup
    istore 0    ; a = 7 * (8 - 3)
    pop

    ; print a;
    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload 0     ; Load a
    invokevirtual java/io/PrintStream/println(I)V

    return
.end method
```

Prolog & Constructor

```
.class public SimpleSample
.super java/lang/Object

.method public <init>()V
    aload_0
    invokenonvirtual java/lang/Object/<init>()V
    return
.end method

.method public static main([Ljava/lang/String;)V
        …
.end method
```

JVM Opcodes

iload <arg>	isub	ifeq <arg>
istore <arg>	imul	goto <arg>
dup		
pop		
ldc <arg>		