Jasmin instructions: Part 1

Csc3210
Yuan Long

Exercise

• Finding errors

; int x=(int)1.0
fconst_1 ; Put a constant 1.0 into stack
istore_1 ; Assign local variable 1 as 1.0
Exercise

• Finding errors

```java
; int x=(int)1.0
fconst_1; Put a constant 1.0 into stack
istore_1; Assign local variable 1 as 1.0
```

Wrong.
fconst_1 is for a float number.
istore_1 is for an integer number.
But they are accessing the same number.
So conversion is needed.

```java
; int x=(int)1.0
fconst_1; Put a constant 1.0 into stack
f2i; Convert a float to integer type
istore_1; Assign local variable 1 as 1.0
```

Exercise

• Finding errors

Demo 1

```java
.class public demo1
.super java/lang/Object
.method public static main([Ljava/lang/String;)V
.limit stack 5

bipush 3
istore_1

getstatic java/lang/System/out Ljava/io/PrintStream;
iload_1
invokevirtual java/io/PrintStream/println(I)V
return
.end method
```

Forgot to define the size of local variable array
Exercise

• Finding errors

Demo 1

```java
.class public demo1
.super java/lang/Object

.method public static main([Ljava/lang/String;)V
 .limit stack 5
 .limit locals 2

    bipush 3
    istore_1

    getstatic java/lang/System/out Ljava/io/PrintStream;
    iload_1
    invokevirtual java/io/PrintStream/println(I)V

.end method
```

Stack size not enough, e.g. change 3 to 6
Excercise

• Write instructions for following Java statements.

• Assume x,y,u,w are in local variable 1, 2, 3, 4 respectively.

```java
x = y - 100;
u = 10;
v = (x + 4)*(y/9 + 11);
w = (x << 5) * (u >> 4);
x = y+'b'+1;
```

Excercise

• Write instructions for following Java statements.

• Assume x,y,u,w are in local variable 1, 2, 3, 4 respectively.

```java
; x = y - 100;
aload_2
bipush 100
isub
istore_1
```
Excercise

• Write instructions for following Java statements.
• Assume x,y,u,w are in local variable 1, 2, 3, 4 respectively.

\[
x = y - 100;
\]
\[
u = 10;
\]
\[
v = (x + 4) \times (y/9 + 11);
\]
\[
w = (x << 5) \times (u >> 4);
\]
\[
x = y^{'b'} + 1;
\]
\[
; u -= 10;
\]
\[
iload_3
\]
\[
bipush 10
\]
\[
isub
\]
\[
istore_3
\]
\[
x = y - 100;
\]
\[
u = 10;
\]
\[
v = (x + 4) \times (y/9 + 11);
\]
\[
w = (x << 5) \times (u >> 4);
\]
\[
x = y^{'b'} + 1;
\]
Excercise

• Write instructions for following Java statements.

• Assume x,y,u,w are in local variable 1, 2, 3, 4 respectively.

```java
x = y - 100;
u = 10;
v = (x + 4)*(y/9 + 11);
w = (x << 3) * (u >> 4);
x = y + 'b' + 1;
```

```java
; (x << 3) * (u >> 4)
iload_1
bipush 3
ishl ; x << 3

iload_3
bipush 4
ishr ; u >> 4
imul
istore 4
```

Excercise

• Write instructions for following Java statements.

• Assume x,y,u,w are in local variable 1, 2, 3, 4 respectively.

```java
x = y - 100;
u = 10;
v = (x + 4)*(y/9 + 11);
w = (x << 5) * (u >> 4);
x = y + 'b' + 1;
```

```java
;x = y + 'b' + 1;
iload_2
ldc “b”
iadd
bipush 1
iadd
istore_1
```
Summary

- **Flow control**
  e.g. if..else, switch(case), goto, for loop, while loop, return

- **Stack**
  e.g. pop, swap, dup
Flow control

• With conditions
  ifXX <label>, if_icmpXX <label>
• Without conditions
  goto <label>

where XX could be replaced by conditions
  eq(==), ne(!=), lt(<), le(<=), gt(>), ge(>=).

ifXX pops one value(integer) from stack and compares it with 0.
if_icmpXX pops two values(integer) from stack and compares them.

Note: No results will be pushed back to stack but program execution
will jump to the instructions associated with label(and will not go
back).

Question 1: Are there any instructions that only compare two numbers and
return the results to the stack?
Question 2: If the condition involves some float numbers or long integers,
what instruction we should use?
  Refer to icmpl, fcmpl, fcmpg, dcmpl, dcmpg.
Flow control

• Comparisons (without branch)
  \texttt{lcmp, fcmpl, fcmpg, dcmpl, dcmpg}
  
  ➢ Pop two values from stack and push back the result (integer).
  ➢ Result: 0 for ==, -1 for <, 1 for > and NaN
  ➢ Prefix \texttt{f} and \texttt{d} mean the type of value should be popped out.
    \texttt{l} - long, \texttt{f} - float, \texttt{d} - double
  ➢ Postfix \texttt{l} and \texttt{g} mean the different treatments on NaN.
    \texttt{l} - treat -1 as NaN, \texttt{g} - treat 1 as NaN.

Flow control

• Comparisons (with no branch)
  \texttt{lcmp, fcmpl, fcmpg, dcmpl, dcmpg}

Pop two values from stack and push back the result (integer).
Result: 0 for ==, -1 for <, 1 for > and NaN

\begin{array}{ccc}
; \text{Long} & ; \text{float} & ; \text{double} \\
\text{integer} & \text{ldc} 3.0 & \text{ldc2\_w} 3.0 \\
\text{ldc2\_w} 3 & \text{ldc} 10.0 & \text{ldc2\_w} 10.0 \\
\text{ldc2\_w} 10 & \text{fcmpl} & \text{dcmpg} \\
lcmp & & \\
\end{array}

Result: -1
Flow control

• With conditions
  ifXX <label>, if_icmpXX <label>
  where XX could be replaced by conditions
  eq(==), ne(! =), lt(<), le(<=), gt(>), ge(>=).

Java
  int x=5,b=3;
  if(x<0) b=-1
    else b=1
    // x in local variable 1
    // b in local variable 2

Jasmin
  ; X<0
  iload_1
  ifge else
    ;b=-1
    iconst_m1
    istore_2
  else:
    ;b=1
    iconst_1
    istore_2
    goto Done

Correct???
No. If the condition is false, after the execution of
istore_2, the instructions labeled by else will still be
executed. E.g. try the example when x=5.

if(x<0) {b=-1,b=1}
  else b=1
Flow control

• With conditions

\( \text{ifXX <label>, if\_icmpXX <label>} \)

where XX could be replaced by conditions

\( \text{eq(==), ne(!=), lt(<), le(<=), gt(>), ge(>)} \).

### Java

```
int x=5, b=3;
if(x<0) b=-1
else b=1
// x in local variable 1
// b in local variable 2
```

### Jasmin

```
; X<0
iload_1
ifge else
; b=-1
iconst_m1
istore_2
goto Done
else:
; b=1
iconst_1
istore_2
Done:
```

### Flow control

• With conditions

For complicated if else java statement, how to write it in Jasmin?

```
if()
...//content 1
else if()
...//content 2
else if()
...//content 3
else if()
...//content 4
else
...// content 5
```

```java
; Load the elements for comparisons
if\_icmpXX else1
;Process on content 1
goto Done
else1:
; Load the elements for comparisons
if\_icmpXX else2
;Process on content 2
goto Done
else2:
; Load the elements for comparisons
if\_icmpXX else3
;Process on content 3
goto Done
else3:
; Load the elements for comparisons
if\_icmpXX else4
;Process on content 4
goto Done
else4:
;Process on content 5
goto Done ; Could be skipped
Done:
```
Flow control

• With conditions

For complicated if else java statement, how to write it in Jasmin?

```java
if(x==1)
   //content 1
else if(x==2)
   //content 2
else if(x==3)
   //content 3
else if(x==4)
   //content 4
else
   //content 5

switch(x){
   case 1:
      //content 1
      break;
   case 2:
      //content 2
      break;
   case 3:
      //content 3
      break;
   case 4:
      //content 4
      break;
   default:
      //content 5
}
```

➢ For this type of if else JAVA statement, besides using if_icmpXX, can we use other instructions to simplify the program in JVM?
➢ The original Java statement can be converted into switch case in Java.
➢ Does JVM have similar switch case instruction? Refer to `tableswitch` and `lookupswitch`

Flow control

• Tableswitch and lookupswitch

➢ Pop one integer value from stack, according to its value jump to different label.

➢ Tableswitch

   Argument: range of values .
   Each label is for one value.
   Labels are ordered from the lowest value and incremented by 1 sequentially until the highest value.

➢ Lookupswitch

   Each label is for one possible value.
Flow control

- Tableswitch and lookupswitch

```
switch(x){
case 2: y=20; break;
case 3: y=30; break;
case 4: y=40; break;
default: y=0;
}
```

```
iload_1
switch 2 4
  case2
  case3
  case4
  defaultCase
    case2: bipush 20 istore_2 goto done
case3: bipush 30 istore_2 goto done
case4: bipush 40 istore_2 goto done
defaultCase: bipush 0 istore_2 goto done
Done:
```

```
iload_1
lookupswitch
  2 : case2
  3 : case3
  4 : case4
default : defaultCase
    case2: bipush 20 istore_2 goto done
case3: bipush 30 istore_2 goto done
case4: bipush 40 istore_2 goto done
defaultCase : bipush 0 istore_2 goto done
Done:
```
Flow control

• while loop and for loop

No while loop or for loop instruction in Jasmin. We need to combine if related instructions, goto and labels in Jasmin.

```java
int sum=0;
int x=1;
while(x<=5)
{
    sum+=x;
    x++;
}
```

```java
iconst_0
istore_1 ;sum=0
iconst_1
istore_2 ;x=1
while:
iload_2
bipush 5
if_icmpgt Done ;x>5
iload_1
iload_2
iadd ; sum+x
istore_1 ; sum+=sum=x
iinc 2 1 ; x++
goto while
Done:
```
Flow control

• While loop and for loop

```java
int sum=0;
for(int x=1;x<=5;x++){
    sum+=x;
}
```

```java
iconst_0
istore_1 ;sum=0
Start:
    iconst_1
    istore_2 ;x=1
    iload_2
    bipush 5
    if_icmpgt Done ; x>5
    iload_1
    iload_2
    iadd ; sum+x
    istore_2 ; sum+=x
    iinc 2 1 ; x++
    goto For
    Done:
```

Flow control

• Return from one method

```java
return, Xreturn(X={i,l,d,f,a})
```

**return**: return type is void

**Xreturn**: pops one value (in type X) from stack and pushes it to operand stack for the invoker method.

Note: After return, all the items on the current method’s operand stack are discarded.
Flow control

• Other
e.g. ret, jsr
Relate to subroutines and calling methods.
You will see them in next class.

Stack operation

• Operations on the stack for current method
  pop, dup, swap
  pop: pops a single word(integer or float) item off stack.
  dup: duplicates a single word(integer or float) item on stack.
  swap: exchanges the top two single word items on stack.

For long integer and double, use pop2, push2, dup2 and swap instead.
Stack operation

- Operations on the stack for current method
  pop, push, dup, swap

Excercise

- What’s the output of following program?

```java
sipush 257
istore_1
Loop:
  iload_1
  ; x==0
  ifeq Done
  ; y=x%2
  iload_1
  iconst_2
  irem
  istore_2
  getstatic java/lang/System/out Ljava/io/PrintStream;
  iload_2
  invokevirtual java/io/PrintStream/println(I)V
  ; x=x/2 or x>>1
  iload_1
  iconst_1
  ishr
  istore_1
  goto Loop
Done:
return
```
Excercise

• Finding errors?

```java
int x = 5, y = 0;
if (x < 3) y = x + 2;
else y = y + 5;
Z = y >> 2;
```

No ',' between 5 and 1.
This makes local variable 5 increment by 1.

ishr has no arguments.
Shift amount should be pushed into stack first.

```
.bipush 5
.istore_1 ; x = 5
.istore_2 ; y = 0

; if (x < 3) y = x + 2; else y = y + 5
.bipush 3
.iif greater

if condition false, and then still go to Done?
The following instructions are ignored.
```

Excercise

• Finding errors?

```java
int x = 5, y = 0;
if (x < 3) y = x + 2;
else y = y + 5;
Z = y >> 2;
```

```
.bipush 5
.istore_1 ; x = 5
.istore_2 ; y = 0

; if (x < 3) y = x + 2; else y = y + 5
.bipush 3
.iif greater
```

Else:
```
iinc 5, 1
Done:
iiload_2
.ishr 2
getstatic java/lang/System/out
.iiload_3
invokevirtual java/io/PrintStream/println(I)V
.return
.end method
```