

Question 1

Not yet
answered

Marked out of
1.50

Flag
question

Let the grammar of while statement as follows:

`while (exp) stmt`

where `stmt` is the body of the while statement, `exp` is the conditional expression.

Assume that `visit(stmt)` is used to generate code for **stmt**, `visit(exp)` for **exp**, `emitIFFALSE(label)` is used generate code to transfer the control to the **label** if the value on stack is **false**, `emitIFTRUE(label)` is used generate code to transfer the control to the **label** if the value on stack is **true**, `emitGOTO(label)` to generate code to unconditionally transfer control to **label** and `emitLABEL(label)` to generate label **label**.

The sequence of code for the while statement is:

`emitLabel(label0)`

`emitLabel(label1)`

`visit(exp)`
`emitIFFALSE(label1)`
`visit(stmt)`
`emitGOTO(label0)`

Question **2**Not yet
answeredMarked out of
1.50Flag
question

Let op be a binary operator like $+$ or $*$ and the binary expression is written as follows:

$exp1 \ op \ exp2$ where $exp1$, $exp2$ are sub expressions and op is a binary operator.

Assume that $visit(op)$ is used to generate code for op and $visit(exp1)$ for $exp1$ and $visit(exp2)$ for $exp2$.

The order of code generation is:

$visit(exp1)$
 $visit(exp2)$
 $visit(op)$

Question 3

Not yet
answered

Marked out of
2.00

Flag
question

Let the grammar of for statement as follows:

for (exp1;exp2;exp3) stmt

where stmt is the body of the while statement, exp1 id the initial expression, exp2 is the conditional expression and exp3 is the increment expression.

Assume that visit(stmt) is used to generate code for stmt, visit(exp) for exp, emitIFFALSE(label) is used generate code to transfer the control to the label if the value on stack is false, emitIFTRUE(label) is used generate code to transfer the control to the label if the value on stack is true, emitGOTO(label) to generate code to unconditionally transfer control to label and emitLABEL(label) to generate label label.

The sequence of code for the while statement is:

emitLabel(label1)

visit(exp1)
emitLabel(label0)
visit(exp2)
emitIFFALSE(label1)
visit(stmt)
visit(exp3)
emitGOTO(label0)

Question 4

Not yet
answered

Marked out of
1.50

Flag
question

Let op be a unary operator like $-$ or $!$ so the prefix unary expression is written as follows:

$op\ exp$ where exp is a sub expression and op is a unary operator.

Assume that $visit(op)$ is used to generate code for op and $visit(exp)$ for exp .

The order of code generation is:

$visit(exp)$

$visit(op)$

Question 5

Not yet
answered

Marked out of
1.50

Flag
question

Let the grammar of do while statement as follows:

do stmt while exp

where stmt is the body of the do while statement, exp is the conditional expression.

Assume that visit(stmt) is used to generate code for **stmt**, visit(exp) for **exp**, emitIFFALSE(label) is used generate code to transfer the control to the **label** if the value on stack is **false**, emitIFTRUE(label) is used generate code to transfer the control to the **label** if the value on stack is **true**, emitGOTO(label) to generate code to unconditionally transfer control to **label** and emitLABEL(label) to generate label **label**.

The sequence of code for the do while statement is:

emitLABEL(label0)
visit(stmt)
visit(exp)
emitIFTRUE(label0)

Question 6

Not yet
answered

Marked out of
1.00

Flag
question

When generating code for **do stmt while exp**, where is the instruction **emitLABEL(breakLabel)** to help generating code for **break** inside **stmt** ?

Select one:

- ☐ a. After the conditional jump instruction to move the control up to **stmt**
- ☐ b. After visit(stmt)
- ☐ c. After visit(exp)
- ☐ d. Before visit(stmt)

A

Question 7

Not yet
answered

Marked out of
1.00

Flag
question

When generating code for `for(exp1;exp2;exp3) stmt`, where is the `emitLABEL(continueLabel)` to help generating code for `continue` instruction inside `stmt` ?

Select one:

- ☐ a. Before visit(exp2)
- ☐ b. Before visit(exp3)
- ☐ c. Before visit(exp1)
- ☐ d. Before visit(stmt)

B