CSci 365: Organizations of Programming Languages

Name: Kenneth Jahnke

**Assignment 5 (chap. 7): 90 points**

Q1. [20] Let the function fun be defined as below and used in the main program:

**int** fun(**int**\* k) {

\*k += 4;

**return** 3 \* (\*k) - 1;

}

**void** main() {

**int** i = 10, j = 10, sum1, sum2;

sum1 = (i/2) + fun(&i);

sum2 = fun(&j) + (j/2);

}

What are the values of sum1 and sum2, assuming

1. [10] Operands in the expressions are evaluated left to right

==============BEGIN ANSWER Q1.1==============

sum1 = (10 / 2) + fun(&i)

sum1 = (5) + fun(&i)

sum1 = 5 + [3 \* (i) - 1]

sum1 = 5 + [3 \* (10) - 1]

sum1 = 5 + [3 \* (14) – 1]

sum1 = 5 + [42 – 1]

sum1 = 5 + [42 - 1]

sum1 = 5 + [41]

**sum1 = 46**

sum2 = fun(&j) + (j / 2)

sum2 = [3 \* (j) - 1] + (j / 2)

sum2 = [3 \* (10) - 1] + (10 / 2)

sum2 = [3 \* (14) - 1] + (14 / 2)

sum2 = [42 - 1] + (14 / 2)

sum2 = [41] + (14 / 2)

sum2 = [41] + (7)

**sum2 = 48**

==============END ANSWER Q1.1==============

1. [10] Operands in the expressions are evaluated right to left.

==============BEGIN ANSWER Q1.2==============

sum1 = (i / 2) + fun(&i)

sum1 = (i / 2) + [3 \* (i) – 1]

sum1 = [(10) / 2] + [3 \* (10) – 1]

sum1 = [(14) / 2] + [3 \* ((14) – 1)]

sum1 = [(14) / 2] + [3 \* 13]

sum1 = [(14) / 2] + [39]

sum1 = [7] + [39]

**sum1 = 46**

sum2 = fun(&j) + (j / 2)

sum2 = fun(&j) + [(10) / 2]

sum2 = fun(&j) + [5]

sum2 = [3 \* (j) - 1] + [5]

sum2 = [3 \* (10) - 1] + [5]

sum2 = [3 \* ((14) - 1)] + [5]

sum2 = [3 \* (13)] + [5]

sum2 = [39] + [5]

**sum2 = 44**

==============END ANSWER Q1.2==============

Q2. [25] Assume the following rules of associativity and precedence for expressions:

|  |  |  |
| --- | --- | --- |
|  |  | Operators |
| Precedence | Highest  Lowest | \*, /, not  +, , &, mod  - (unary)  =, /=, <. <=, >=, >  and  or, xor |
| Associativity | Left to Right |  |

Show the order of evaluation of the following expressions by parenthesizing all subexpressions and placing a superscript on the right parenthesis to indicate order.

For example, for the expression

a + b \* c + d

the order of evaluation would be represented as

((a + (b \* c)1)2 + d)3

1. [7] (a - b) / c & (d \* e / a – 3)

==============BEGIN ANSWER Q2.1==============

(((a - b)1 / c)5 & (((d \* e)2 / a)3 – 3)4)6

==============END ANSWER Q2.1==============

1. [6] -a or c = d and e

==============BEGIN ANSWER Q2.2==============

((-a)1 or ((c = d)2 and e)3)4

==============END ANSWER Q2.2==============

1. [6] a > b xor c or d <= 17

==============BEGIN ANSWER Q2.3==============

(((a > b)1 xor c)3 or (d <= 17)2)4

==============END ANSWER Q2.3==============

1. [6] -a + b

==============BEGIN ANSWER Q2.4==============

(-(a + b)1)2

==============END ANSWER Q2.4==============

Q3. [15]

Write a BNF description of the precedence and associativity rules defined for the expressions in Q2. Assume the only operands are the names a, b, c, d, and e, and <expr> is the start non-terminal for the expression.

Use the following non-terminal and terminal symbols to define the rules in BNF.

Non-terminals: <expr>, <e1>, <e2>, <e3>, <e4>, <e5>, etc. ,

where <eq*n*> has a higher hierarchy than <eq*n+1*>, that can be defined using <eq *n+1*> or lower. e.g.) <eq2>  <e2> + <e3>, <e3>  <e3> / <e4>, etc.

Terminals: a, b, c, d, e

Operators: all the operators given in the table of Q3.

==============BEGIN ANSWER Q3==============

<expr> ::= <expr> “or” <e1> | <expr> “xor” <e1> | <e1>

<e1> ::= <e1> “and” <e2> | <e2>

<e2> ::= <e2> “=” <e3> |

<e2> “/=” <e3> |

<e2> “<” <e3> |

<e2> “<=” <e3> |

<e2> “>=” <e3> |

<e2> “>” <e3> |

<e3>

<e3> ::= “-” <e3> | <e4> // unary minus

<e4> ::= <e4> “+” <e5> |

<e4> “-” <e5> | // subtraction minus

<e4> “&” <e5> |

<e4> “mod” <e5> |

<e5>

<e5> ::= <e5> “\*” <e6> | <e5> “/” <e6> | <e5> “not” <e6> | <e6>

<e6> ::= <e7> | <id>

<e7> ::= “(” <expr> “)”

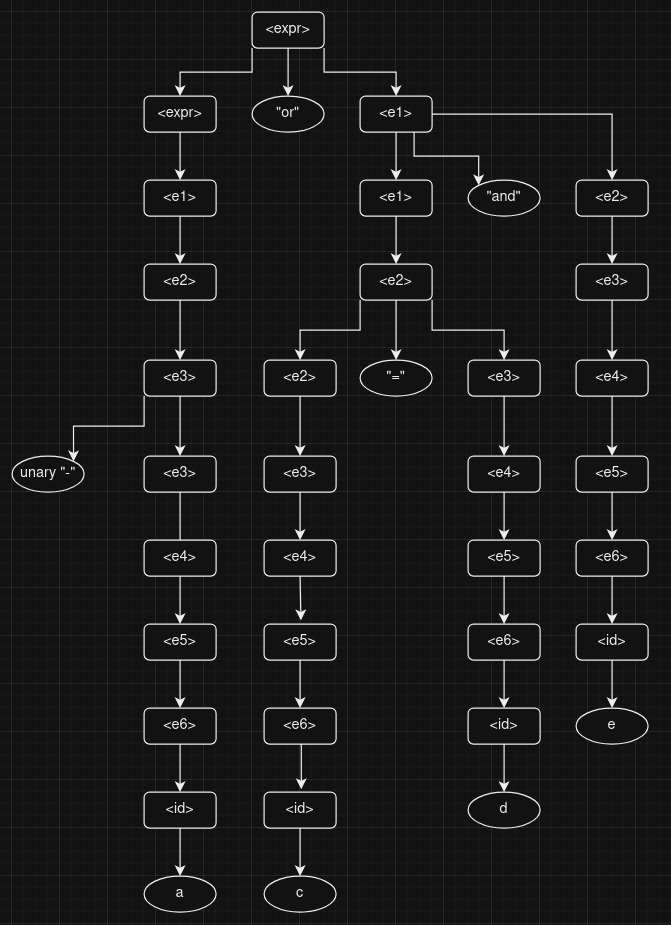
<id> ::= a | b | c | d | e | <constant>

==============END ANSWER Q3==============

Q4. [30] Using the grammar in Q3, draw a parse tree for the expression.

1. [10] -a or c = d and e

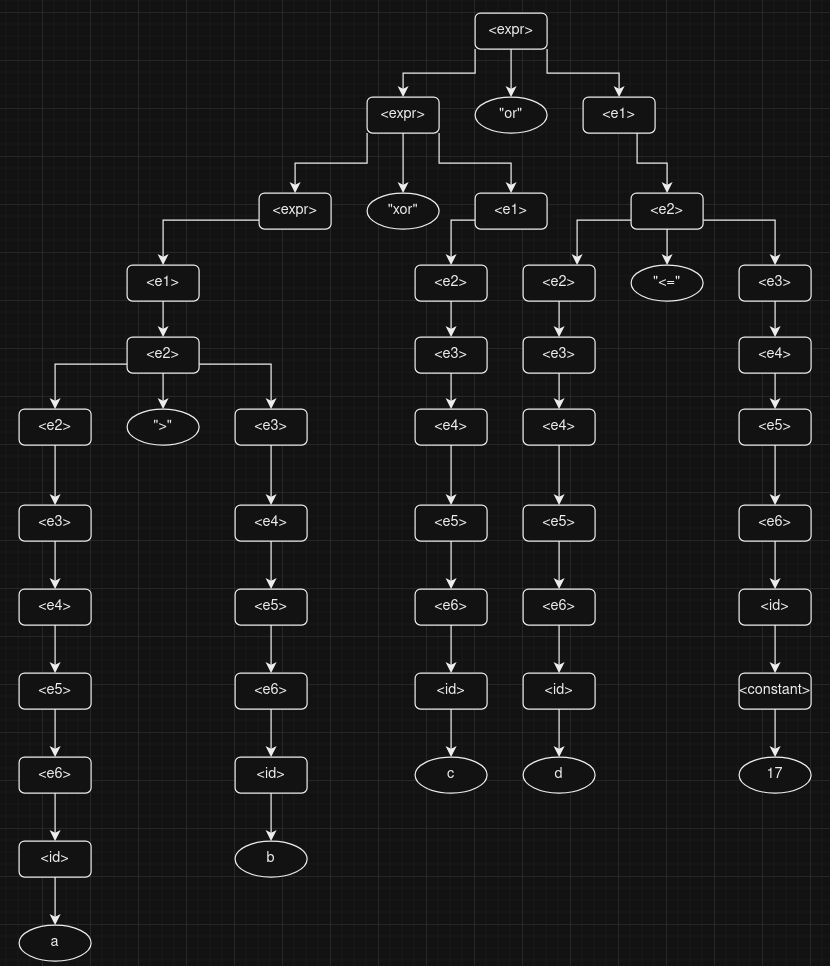
==============BEGIN ANSWER Q4.1==============



==============END ANSWER Q4.1==============

1. [10] a > b xor c or d <= 17

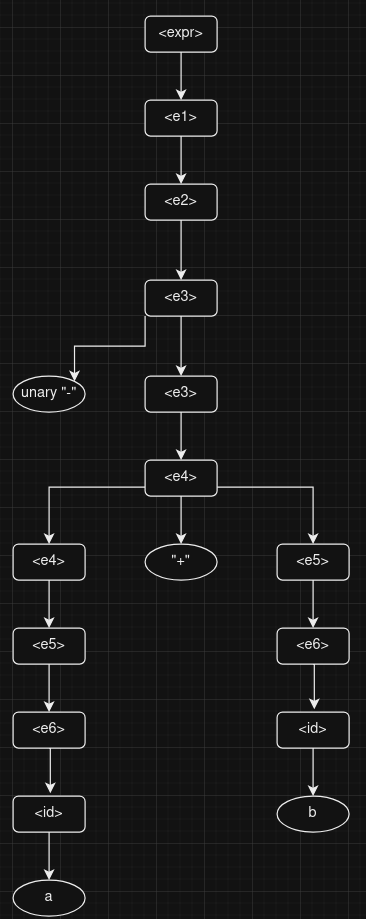
==============BEGIN ANSWER Q4.2==============



==============END ANSWER Q4.2==============

1. [10] -a + b

==============BEGIN ANSWER Q4.3==============



==============END ANSWER Q4.3==============