Brian Nguyen

Assignment 1

**(1.1.3 Section Review)**

1. Assemblers are programs that take code written in assembly language and translate that code into machine code. The Linker is the program that takes the translated files produced by the Assembler and combines them together to create a single executable file.

3. A one-to-many relationship is the fact that an instruction in a high-level language when translated to machine language produces many more instructions.

8 Type checking on pointer variables is probably stricter in C and C++ than in assembly language as a main idea of high-level language is to create more rules to prevent producing logic errors.

12.

mov eax,Y

mov ebx,4

imul ebx

add eax,3

mov X,eax

**(1.2.1 Section Review)**

2. Translated programs might execute more quickly than interpreted programs because the higher level language in translated programs are all translated and ready to be executed while interpreted programs have to wait for each instruction to be translated in order for those instructions to be executed.

3. True

10. Statements in Assembly Language (Level 3) are translated into statements into the Instruction Set Architecture/ machine language (level 2).

**(1.3.9 Section Review)**

2a. 248 b. 202 c. 240

3a. 00010001 b. 01000000 c.00011110

5a. 7 b. 9 c. 15

6a. 35DA b. CEA3 c. FEDB

7a. 1010 0100 1010 1001 0011 1111 1011 1100

b. 1011 0110 1001 0111 1100 0111 1010 0001

c. 0010 1011 0011 1101 1001 0100 0110 0001

**(1.7.1 Short Answer)**

1. The leftmost digit.

2a. 53 b. 150 c. 204

3a. 10001010 b. 10010110 c. 00100001

4. 00000110

5a. word: 16 bits b. doubleword: 32 bits c. quadword: 64 bits d. double quadword: 128 bits

6a. 12 b. 16 c. 16

7a. 35DA b. CEA3 c. FEDB

8a. 0000 0001 0010 0110 1111 1001 1101 0100

b. 0110 1010 1100 1101 1111 1010 1001 0101

c. 1111 0110 1001 1011 1101 1100 0010 1100

9.a 58 b. 447 c. 4097

10a. 98 b. 1203 c. 671

11a. E8 b. EB5

12a. EC b. D3

13a. 27641 b. -16093

14a. 19666 b. -32208

15a. -75 b. 42 c. -16

16a. -128 b. -52 c. -73

17a. 11111011 b. 11010110 c. 11110000

18a. 10111000 b. 10011110 c. 11100110

19a. B82 b. 1106

20a. B82 b. 1316

21. Hexadecimal: 42 Decimal: 66

22. Hexadecimal: 47 Decimal: 71

25.

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | A v B | ¬ (A v B) |
| F | F | F | T |
| F | T | T | F |
| T | F | T | F |
| T | T | T | F |

**(1.7.2 Algorithm Workbench)**

3. Write a function that receives an integer. The function must return a string containing the

binary representation of the integer.

(In C++)

string getBinaryString(int decimalValue)

{

int quotient = abs(decimalValue);

int remainder;

string binaryValueString;

while (quotient > 0)

{

remainder = quotient % 2;

quotient = quotient / 2;

binaryValueString.insert(0, std::to\_string(remainder));

}

return binaryValueString;

}

**EXTRA CREDIT: (1.4.2 Section Review)**

1.

|  |  |  |  |
| --- | --- | --- | --- |
| X | Y | ¬X | ¬X v Y |
| F | F | T | T |
| F | T | T | T |
| T | F | F | T |
| T | T | F | T |

In all cases the Boolean expression ¬X v Y produces True.

2.

|  |  |  |
| --- | --- | --- |
| X | Y | X ^ Y |
| F | F | F |
| F | T | F |
| T | F | F |
| T | T | T |

The “and” in the Boolean expression X ^ Y make it so the only case where the output is true is when both X and Y are true.

3. True

4. False

5. True