NAME: Kenneth Robinson CS466-001

CHAPTER: 6

LAB: Lab 5

ANIMATED FLASHCARDS

1. Abstract Step

2. Algorithm

3. Assignment statement

4. Base class

5. Brainstorming

6. Class (general sense)

7. Client

8. Concrete step

9. Container class

10. Control abstraction

11. Control structure

12. CRC Cards

13. Data abstraction

14. Desk checking

15. Encapsulation

16. Fields

17. Filtering

18. Heuristics

19. Identifier

20. Implementation phase

21. Information hiding

22. Inspection

23. Maintenance phase

24. Method

25. Module

26. Object

27. Object class (class)

28. Object-oriented design

29. Problem solving

30. Problem-solving phase

31. Procedural abstraction

32. Program

33. Programming language

34. Pseudocode

35. Responsibility algorithms

36. Reuse

37. Scenarios

38. Semantics

39. Sort Key

40. Sorting

41. Syntax

42. Top-down design

43. Walk-through

BOOK EXERCISES

1. B

2. A

3. A

4. A

5. B

6. B

7. A

8. A

9. B

10. A

11. B

12. A

13. B

14. B

15. A

16. A

17. D

18. C

19. B

20. E

21. Programmable means that data and instructions are logically the same

and are stored in the same place. The consequence of this fact is that

the program the computer executes is not wired into the hardware but

entered from outside.

22. There must be machine-language instructions to store, retrieve, and

process data, to input data, and to output data. These instructions

mirror the operations of the von Neumann machine

23. A machine language instruction can perform only one low-level task..

29. The IR contains an instruction (the one being executed); the PC contains an address (the address of the next instruction to be executed).

33. A is written on the screen.

34. 7

41. One byte of storage is set up for each input value.  If the value that is read is grater than one byte, the excess spills over to the byte above giving the wrong answer.

42. The test plan gives the correct answers with this code.

43. The answers are correct

44.

Read num1

Read num2

Read num3

Load num1

Add num3

Sub num2

Store in answer

Write answer

45.

**BR                Main**

**sum:                .WORD                0x0000**

**num1:               .BLOCK               2 num2:**

**.BLOCK               2**

**num3:               .BLOCK               2**

**answer:.BLOCK              2 Main:**

**DECI num1,d**

**DECI num2,d**

**DECI num3,d**

**LDA  num1,d**

**ADDA num3,d**

**SUBA num2,d**

**STA answer,d**

**DECO answer,d**

**STOP**

**.END**

46.

|  |  |  |  |
| --- | --- | --- | --- |
| *Reason for Test Case* | *Input Values* | *Expected Output* | *Observed Output* |
| Assumption: Input values are no greater than 215 -1 or less than – 215. | | | |
| Input three positive numbers | 4, 6, 1 | -1 | -1 |
| Input three negative numbers | -4, -6, -1 | 1 | 1 |
| Input mixed numbers | 4, 6, -1  4, -6, 1  4, -6, -1  -4, 6, 1  -4, 6, -1  -4, -6, 1 | -3  11  -9  -9  -11  3 | -3  11  -9  -9  -11  -3 |
| Large numbers | 32767, -1, +1 | overflows | overflows |
|  | 32767, 1, -1 | 32767 |  |
|  |  |  |  |
|  |  |  |  |

47.

Set count to 0

Set sum to 0

WHILE count < 4

              Read num

              Add num to sum

              Set count to count + 1

Write sum

              BR                Read              ;branch to location Read

sum:              .WORD  0x0000              ;set word to zero

count: .WORD 0x0000              ;set up a two byte block for count               limit:               .WORD 0x0004     ;set up a block for value 4       number:.BLOCK 2         ;set up word for value read

Read:              LDA    sum,d

              DECI   number,d

              ADDA              number,d

              STA    sum,d              ;store accumulator in sum

              LDA              count,d              ;load a copy of limit in accumulator                             ADDA             1,i

              STA   count,d              ;store contents in count

              CPA              limit,d              ;compare accumulator to 0

              BREQ              Quit                    ;branch to Quit if count is 4

              BR              Read                            ;go back to read in another number

Quit:              DECO              sum,d

              STOP

              .END

48. A data-coverage plan is written without looking at the code, so the same test plan would be valid.  A code-coverage plan looks at the code, but because there is a one to one relationship between a machine code instruction and an assembly language instruction, the same test plan would be valid.

55.

Write "Enter three integer values"

Read first, second, third

IF (first < second)

              IF (second < third)

                            Write first, second, third

              ELSE

                            IF (first < third)

                                          Write first, third, second

                            ELSE

                                          Write third, first, second

ELSE

              IF (first < third)

                            Write second, first, third

              ELSE

                            IF (second < third)

                                          Write second, third, first

                            ELSE

                                          Write third, second, first

56.

Write "Enter three values; a negative value stops the processing"

Read first, second, third

WHILE (first > 0)

              // Process the three values

              Write "Enter three values; a negative value stops the processing"

              Read first, second, third

57.

Write "Enter three values; a negative value stops the processing"

Read first

WHILE (first > 0)

              Read second, third

              // Process the three values

              Write "Enter three values; a negative value stops the processing"

              Read first

58. Pseudocode is a language for expressing algorithms.  Pseudo-operations are instructions to a translating system.

59. Variables, assignment, input/output, repetition, selection (if-then, if-then-else)

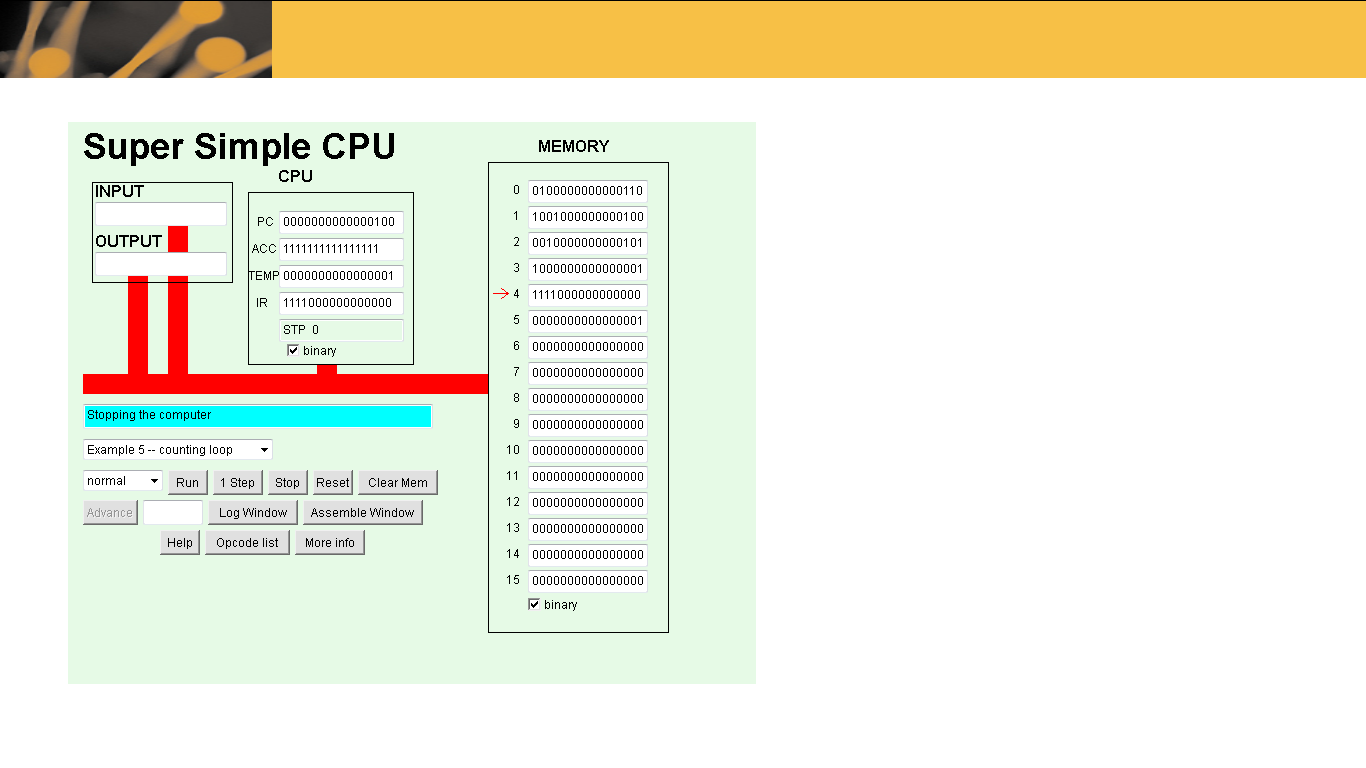
60. A looping construct repeats an action (s) while a condition is true.  A selection construct determines if a condition is true and does one thing if it is and another thing if it is not.

LAB EXERCISES

**Exercise 4**

1. 0000000000000001, 0000000000000010, 0000000000000011, 0000000000000100

4.



5. Cell 0: LDI into cell 6

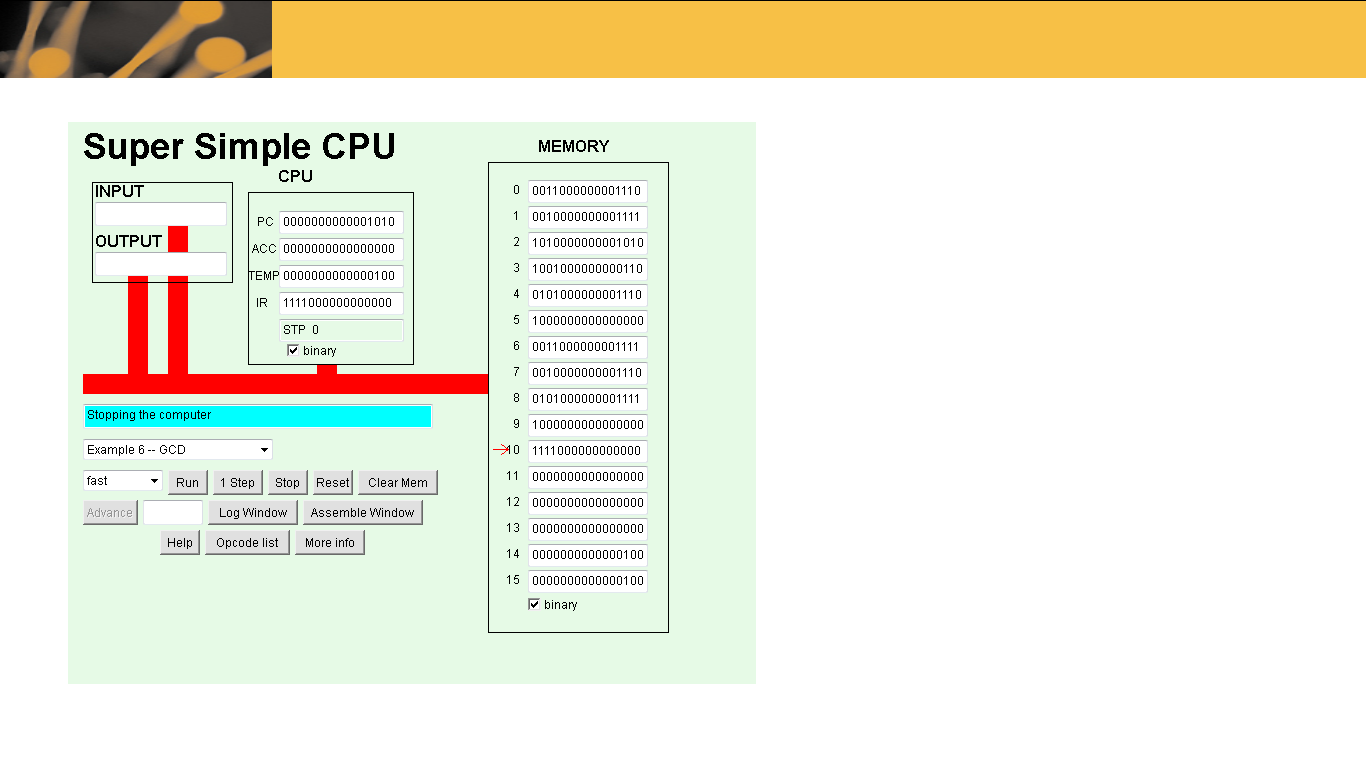
Cell 1: JNG Jump to instruction in cell 4 if accumulator <0

Cell 2: SUB Subtract operand from accumulator in cell 1

Cell 3: JNP Jump to instruction in cell 1

Cell 4: STP counter

**Exercise 5**

****

**Exercise 6**

0 0010000000001101

1 0011000000001101

2 0101000000001100

3 1111000000000000