Last name (Surname, family name) _____
First name (Given name) _____
Student ID _____

# Read this first!

- Final exam of CISP310 (2018/05/15)
- Time: 2000 to 2200
- Location: Raef 161
- Open material printed or hand written prior to the exam, with the exception of the final exam itself of the same semester, regardless of class section.
- Individual assessment:
  - The score should reflect the competency of the student with the name and ID as indicated above without any prior knowledge of the exam questions.
  - Answer should be independently and originally derived in the allotted time frame, observing all applicable rules and restrictions.
  - No sharing of open book/notes material during the exam.
  - Do not discuss the questions with students who may not have taken the exam.
- Use additional sheets of paper if necessary
  - Write your name on each extra sheet of paper.
  - Clearly indicate the question number.
- Write legibly, I cannot give points to answers that I cannot interpret without post exam explanation.
- Make your answers as complete as possible. If there is something that I look for to assign partial credit, I cannot and will not infer that from what you write. In any proof or trace, write your answers as completely as possible.
- All questions have equal weight.
- Some questions may cross pages, be sure to read and answer the entire question!
- The exam accounts for 40% of your final grade.

## Important!

Use the 7 digits of your student ID to fill member "v" of the structures in listing 1 and listing 2. "n1" corresponds to the leftmost digit, while "n7" corresponds to the rightmost digit.

Listing 2 is the TTPASM implementation of listing 1 in C. The actual definition of "traverse" is the big mystery of this exam.

If a question asks for explanation of assembly language code, the explanations should relate assembly language code to C language constructs.

If a memory location is updated, explain what is updated. For example "`*param1 = 0`" if what param1 points to is initialized to 0.

If an explanation is below C construct level, relate to stack content and operations such as "`push param1+1`" or "`pop saved return value of f(x,y) to c`".

If a line updates a register, explain what the new value is relating it to C concepts. Use the typical notation that I use like "`b==&(ptr->value)`". If the explanation is below C construct level, use stack related explanations such as "dealloc local variables".

If a line changes flow of execution, explain what that does and relate the change of flow of execution to structured C programming concepts. Where is the branch destination? (Such as '`go to beginning of while loop`" or "call function `mysterio`") Is it conditional? If so, when are we going? (Such as "`exit loop iff i<j`".)

If a line has a side effect on flags that is used later, explain what the instruction does and relate it to the use of the flag(s), such as "`update L flag that is used in condtional branch`".

Listing 1: C code listing

```
1  #include <stdint.h>
2  struct X
3  {
4    struct X *l;
5    uint8_t v;
6    struct X *r;
7  };
8  extern struct X n1, n2, n3, n4, n5, n6, n7;
9  struct X n1 {&n2, ?, &n3};
10 struct X n2 {&n4, ?, &n5};
11 struct X n3 {&n6, ?, &n7};
12 struct X n4 {0, ?, 0};
13 struct X n5 {0, ?, 0};
14 struct X n6 {0, ?, 0};
15 struct X n7 {0, ?, 0};
16
17 int traverse(struct X *ptr)
18 {
19   // what goes here?
20 }
21
22 int main()
23 {
24   uint8_t x;
25   x = traverse(n?);
26   return 0;
27 }
```

Listing 2: TTPASM code listing

```
1  sub  d,d
2  jmpi main
3
4  n1:
5    byte n3
6    byte ?
7    byte n2
8  n2:
9    byte n4
10   byte ?
11   byte n5
12 n3:
13   byte n6
14   byte ?
15   byte n7
16 n4:
17   byte 0
18   byte ?
19   byte 0
20 n5:
21   byte 0
22   byte ?
23   byte 0
24 n6:
25   byte 0
26   byte ?
27   byte 0
28 n7:
29   byte 0
30   byte ?
31   byte 0
32
33 main:
34   dec d
35   ldi a,n?
36   dec d
37   st  (d),a
38   ldi a,mainRetFromTraverse
39   dec d
40   st  (d),a
41   jmpi traverse
42 mainRetFromTraverse:
43   inc d
44   st  (d),a
45   ldi a,0
46   inc d
47   halt
48
49 traverse:
50   cpr c,d
51   inc c
52   ld  a,(c)
53   and a,a
54   jzi L1
55   dec d
56   st  (d),a
57   ld  a,(a)
58   dec d
59   st  (d),a
60   ldi a,L3
61   dec d
62   st  (d),a
63   jmpi traverse
64 L3:
65   inc d
66   ld  b,(d)
67   add a,a
68   st  (d),a
69   inc b
70   inc b
71   ld  b,(b)
72   dec d
73   st  (d),b
74   ldi a,L4
75   dec d
76   st  (d),a
77   jmpi traverse
78 L4:
79   inc d
80   ld  b,(d)
81   inc d
82   add a,b
83   cpr c,d
84   inc c
85   ld  b,(c)
86   inc b
87   ld  b,(b)
88   add a,b
89   jmpi L2
90 L1:
91   ldi a,0
92 L2:
93   ld  b,(d)
94   inc d
95   jmp b
```

## Question 1 of 6

Identify the lines in listing 2 that implement line 25 in listing 1.

In the following space, copy those lines including the line numbers, then comment and explain what each does according to what is outlined in the section labeled "Important".

## Question 2 of 6

Identify the overall control structure of the subroutine "traverse" in listing 2. Specify the control structure and if it involves any condition, specify the condition as well. There is no need specify individual statements nested in the control structure.

In the following space, copy the lines from listing 2 that relate to the control structure, including the line number, and instruction mnemonic. Comment and explain what each does according to what is outlined in the section labeled "Important".

Note that the lines for this question need not be contiguous. If a range of lines does not relate to a control structure, they should be left out. Control structures include loops, conditional statements and return statements, however, return values are not considered a part of the control structure.

## Question 3 of 6

The following is a section of code in listing 2, lines 55 to 65.

Comment and explain what each does according to what is outlined in the section labeled "Important".

If a register or stack content is initialized prior to this section of code, indicate so and explain the value of the register or stack content prior to this section of code in high level C code concepts.

```
55    dec d
56    st  (d),a
57    ld  a,(a)
58    dec d
59    st  (d),a
60    ldi a,L3
61    dec d
62    st  (d),a
63    jmpi traverse
64  L3:
65    inc d
```

## Question 4 of 6

The following is a section of code in listing 2, lines 66 to 82.

Comment and explain what each does according to what is outlined in the section labeled "Important".

If a register or stack content is initialized prior to this section of code, indicate so and explain the value of the register or stack content prior to this section of code in high level C code concepts.

```
66    ld  b,(d)
67    add a,a
68    st  (d),a
69    inc b
70    inc b
71    ld  b,(b)
72    dec d
73    st  (d),b
74    ldi a,L4
75    dec d
76    st  (d),a
77    jmpi traverse
78  L4:
79    inc d
80    ld  b,(d)
81    inc d
82    add a,b
```

## Question 5 of 6

Based on listing 2, decompile the subroutine "traverse" in the following space. In other words, write the structured (no goto, break or continue) C version of the function in the following space. Your answer will be graded based on consistency with earlier questions, syntax and completeness. Note that the earlier questions only represent some pieces of the actual code, you will need to fill in more details to answer this question.

## Question 6 of 6

According to listing 2, what are the return values of the following calls based on the structures initialized with digits of your student ID?

Explain your answer briefly. Regardless of your answers to earlier questions, you can use English explanation here. A random guess without an explanation does not get any points!

Do not assume each sub-question carries the same weight!

- x=traverse(0)
- x=traverse(&n6)
- x=traverse(&n3)
- x=traverse(&n1)