Results

Survey 133581

Number of records in this query:	37
Total records in survey:	37
Percentage of total:	100.00%

Field summary for BQ1

What is your highest level of education?

Answer	Count	Percentage
School Graduation (A1)	1	2.70%
Bachelor (A2)	22	59.46%
Master (A3)	12	32.43%
PhD (A4)	2	5.41%
No answer	0	0.00%

Field summary for BQ2

How often do you reverse malware?

Answer	Count	Percentage
Monthly (A3)	5	13.51%
Weekly (A2)	2	5.41%
Daily (A1)	1	2.70%
Less (A4)	29	78.38%
No answer	0	0.00%

Field summary for BQ3

For how long are you reversing malware?

Answer	Count	Percentage
>5 years (A1)	5	13.51%
3 years (A3)	1	2.70%
2 years (A4)	3	8.11%
(A5)	28	75.68%
No answer	0	0.00%

Field summary for BQ4(SQ001)

How much experience do you have in .. ? [Reversing]

Answer	Count	Percentage
1 (1)	4	10.81%
2 (2)	8	21.62%
3 (3)	8	21.62%
4 (4)	3	8.11%
5 (5)	5	13.51%
6 (6)	3	8.11%
7 (7)	2	5.41%
8 (8)	2	5.41%
9 (9)	1	2.70%
10 (10)	1	2.70%
No answer	0	0.00%

Field summary for BQ4(SQ002)

How much experience do you have in .. ? [Malware Analysis]

Answer	Count	Percentage
1 (1)	14	37.84%
2 (2)	8	21.62%
3 (3)	3	8.11%
4 (4)	1	2.70%
5 (5)	3	8.11%
6 (6)	3	8.11%
7 (7)	3	8.11%
8 (8)	1	2.70%
9 (9)	0	0.00%
10 (10)	1	2.70%
No answer	0	0.00%

Field summary for BQ4(SQ003)

How much experience do you have in .. ? [Writing C-code]

Answer	Count	Percentage
1 (1)	1	2.70%
2 (2)	1	2.70%
3 (3)	7	18.92%
4 (4)	7	18.92%
5 (5)	4	10.81%
6 (6)	4	10.81%
7 (7)	3	8.11%
8 (8)	4	10.81%
9 (9)	5	13.51%
10 (10)	1	2.70%
No answer	0	0.00%

Field summary for BQ5

Which binary code decompiler have you used before? (multiple answers are possible)

Answer	Count	Percentage
Hex-Rays (SQ001)	25	67.57%
Ghidra (SQ002)	21	56.76%
BinaryNinja HLIL (SQ003)	3	8.11%
Boomerang (SQ004)	0	0.00%
REC (SQ005)	3	8.11%
DISC (SQ006)	0	0.00%
Other	6	16.22%

ID	Response
37	JD-GUI
78	nil
81	ida
94	none
107	OLLYDEBUG
108	IDA

Field summary for arandomsample

Answer	Count	Percentage
Answer	37	100.00%
No answer	0	0.00%

ID	Response
37	1
40	2
43	1
49	2
51	1
55	1
58	3
61	1
64	2
65	3
68	3
69	2
70	1
75	3
77	2
78	1
79	2
80	1
81	1
82	1
83	3
84	2
89	2
93	1
94	1
95	2
96	2
97	1
103	1
107	3
108	2
109	2
113	3
114	3
116	3
117	2
120	1

Field summary for Aour(1a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The used control-flow structures are appropriate.]

Answer	Count	Percentage
strongly disagree (A1)	1	6.67%
disagree (A2)	1	6.67%
weakly disagree (A3)	1	6.67%
unsure (A4)	3	20.00%
weakly agree (A5)	3	20.00%
agree (A6)	5	33.33%
strongly agree (A7)	1	6.67%
No answer	0	0.00%

Field summary for Aour(1b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The control-flow is strangely restructured.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	4	26.67%
weakly disagree (A3)	2	13.33%
unsure (A4)	2	13.33%
weakly agree (A5)	3	20.00%
agree (A6)	2	13.33%
strongly agree (A7)	2	13.33%
No answer	0	0.00%

Field summary for Aour(3a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [It was easy to understand the code.]

Answer	Count	Percentage
strongly disagree (A1)	1	6.67%
disagree (A2)	4	26.67%
weakly disagree (A3)	3	20.00%
unsure (A4)	4	26.67%
weakly agree (A5)	1	6.67%
agree (A6)	2	13.33%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(3b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [It took much effort to understand the code.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	6.67%
weakly disagree (A3)	3	20.00%
unsure (A4)	2	13.33%
weakly agree (A5)	4	26.67%
agree (A6)	3	20.00%
strongly agree (A7)	2	13.33%
No answer	0	0.00%

Field summary for Aour(4a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [It seems that there are no unused instructions.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	2	13.33%
weakly disagree (A3)	1	6.67%
unsure (A4)	1	6.67%
weakly agree (A5)	3	20.00%
agree (A6)	5	33.33%
strongly agree (A7)	3	20.00%
No answer	0	0.00%

Field summary for Aour(4b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [There are too much unused instructions.]

Answer	Count	Percentage
strongly disagree (A1)	3	20.00%
disagree (A2)	1	6.67%
weakly disagree (A3)	8	53.33%
unsure (A4)	0	0.00%
weakly agree (A5)	1	6.67%
agree (A6)	1	6.67%
strongly agree (A7)	1	6.67%
No answer	0	0.00%

Field summary for Aour(5a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [I think the code is correctly decompiled.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	6.67%
weakly disagree (A3)	0	0.00%
unsure (A4)	8	53.33%
weakly agree (A5)	2	13.33%
agree (A6)	4	26.67%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(5b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The decompiled code seems to be incorrect.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	20.00%
weakly disagree (A3)	3	20.00%
unsure (A4)	7	46.67%
weakly agree (A5)	0	0.00%
agree (A6)	1	6.67%
strongly agree (A7)	1	6.67%
No answer	0	0.00%

Field summary for Aour(6a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The conditions are too complex.]

Answer	Count	Percentage
strongly disagree (A1)	2	13.33%
disagree (A2)	8	53.33%
weakly disagree (A3)	1	6.67%
unsure (A4)	0	0.00%
weakly agree (A5)	3	20.00%
agree (A6)	1	6.67%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(6b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The code contains too many intermediate results.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	20.00%
weakly disagree (A3)	3	20.00%
unsure (A4)	4	26.67%
weakly agree (A5)	3	20.00%
agree (A6)	2	13.33%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(6c)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The line length is too long]

Answer	Count	Percentage
strongly disagree (A1)	8	53.33%
disagree (A2)	6	40.00%
weakly disagree (A3)	1	6.67%
unsure (A4)	0	0.00%
weakly agree (A5)	0	0.00%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(7a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [Many variables have useless copies.]

Answer	Count	Percentage
strongly disagree (A1)	1	6.67%
disagree (A2)	4	26.67%
weakly disagree (A3)	5	33.33%
unsure (A4)	1	6.67%
weakly agree (A5)	2	13.33%
agree (A6)	1	6.67%
strongly agree (A7)	1	6.67%
No answer	0	0.00%

Field summary for Aour(7b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [There are no useless copies of variables.]

Answer	Count	Percentage
strongly disagree (A1)	1	6.67%
disagree (A2)	1	6.67%
weakly disagree (A3)	3	20.00%
unsure (A4)	0	0.00%
weakly agree (A5)	4	26.67%
agree (A6)	5	33.33%
strongly agree (A7)	1	6.67%
No answer	0	0.00%

Field summary for Aour(8a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [The variable types seem to be reasonable.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	6.67%
weakly disagree (A3)	4	26.67%
unsure (A4)	1	6.67%
weakly agree (A5)	4	26.67%
agree (A6)	5	33.33%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(8b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [Some variable types seem to be wrong.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	20.00%
weakly disagree (A3)	3	20.00%
unsure (A4)	3	20.00%
weakly agree (A5)	4	26.67%
agree (A6)	2	13.33%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for Aour(9a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [There are no variables that only store unnecessary intermediate constants.]

Answer	Count	Percentage
strongly disagree (A1)	2	13.33%
disagree (A2)	2	13.33%
weakly disagree (A3)	0	0.00%
unsure (A4)	1	6.67%
weakly agree (A5)	4	26.67%
agree (A6)	5	33.33%
strongly agree (A7)	1	6.67%
No answer	0	0.00%

Field summary for Aour(9b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). .tb_button {padding:1px;cursor:pointer;border-right: 1px solid #8b8b8b;border-left: 1px solid #FFF;border-bottom: 1px solid #fff;}.tb_button.hover {borer:2px outset #def; background-color: #f8f8f8 !important;}.ws_toolbar {z-index:100000} .ws_toolbar .ws_tb_btn {cursor:pointer;border:1px solid #555;padding:3px} .tb_highlight{background-color:yellow} .tb_hide {visibility:hidden} .ws_toolbar img {padding:2px;margin:0px} [There are too many variables that just store intermediate constants.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	4	26.67%
weakly disagree (A3)	4	26.67%
unsure (A4)	2	13.33%
weakly agree (A5)	5	33.33%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AourQ1(SQ001)

What is the output of the function for the parameters [arg_1 = 0 and arg_2 = 1]

Answer	Count	Percentage
0 (A1)	0	0.00%
1 (A2)	0	0.00%
2 (A3)	14	93.33%
3 (A4)	0	0.00%
4 (A5)	0	0.00%
5 (A6)	0	0.00%
I do not know (A7)	1	6.67%
No answer	0	0.00%

Field summary for AourQ1(SQ002)

What is the output of the function for the parameters [arg_1 = 1 and arg_2 = 0]

Answer	Count	Percentage
0 (A1)	0	0.00%
1 (A2)	0	0.00%
2 (A3)	13	86.67%
3 (A4)	0	0.00%
4 (A5)	0	0.00%
5 (A6)	0	0.00%
I do not know (A7)	2	13.33%
No answer	0	0.00%

Field summary for AourQ2

Which computation needs more recursive calls?

Answer	Count	Percentage
arg_1 = 1 and arg_2 = 2 (A1)	4	26.67%
arg_1 = 2 and arg_2 = 1 (A2)	11	73.33%
No answer	0	0.00%

Field summary for AourQ4

What do you like or dislike about the above sample?

Answer	Count	Percentage
Answer	10	66.67%
No answer	5	33.33%

ID	Response
37	I'm not sure what the function was intendend to do, hence, I also don't know if it is decompiled correctly. Also, I'm not sure whether the code could have been simplified. I think so, but I would have to spend more time to think about it. Still, the code was easy and understandable enough to follow it using pen and paper. What I didn't like (major issue): the "+ 0xffffffff" and "& 0xffffffff" parts. I'm not sure if the 0xffffffff in line 9 does anything and it took me really long to figure out that "+ 0xffffffff" is -1 (you got me here). Especially the 0xffffffff in line 21 is totally confusing and should definitely be -1. What I didn't like (minor issue): I'm not a fan of casts cluttering the code. I removed them in the editor to not be distracted anymore. I'm not sure if there are corner cases, but do we really need them in this function? What I didn't like (even less important): I would have preferred an early return in line 5. (Why do I have to answer the each question twice (pos./neg. form)?)
43	Recursion. Bah.
51	var_0 seems reused for a lot of things
55	untidy while loop
70	Simple computations
78	no comments
80	Too many if statement. Not easy to understand what the code trying to do
81	types are not normalized in stmt
93	1.) this is clearly obfuscation and written to make it most hard for the reader/reverser in all aspects! 2.) Also, the *author* might *not* have considered that this code would produce quite different result on different OS, such as Windows, to name an important OS. ;-) On Windows, an signed/unsigned int == 64 bit, on a 64 bit OS. Thus, it would NOT lead to overflow with 1 + 0xffffffff. We did assume a Linux system here, where a signed/unsigned int == 32 bit register size, leading to overflow with 1 + 0xffffffff.
120	There is no recursion

Field summary for AIDA(1a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The used control-flow structures are appropriate.]

Answer	Count	Percentage
strongly disagree (A1)	1	7.69%
disagree (A2)	0	0.00%
weakly disagree (A3)	2	15.38%
undecided (A4)	6	46.15%
weakly agree (A5)	2	15.38%
agree (A6)	2	15.38%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(1b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The control-flow is strangely restructured.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	7.69%
weakly disagree (A3)	3	23.08%
undecided (A4)	3	23.08%
weakly agree (A5)	4	30.77%
agree (A6)	1	7.69%
strongly agree (A7)	1	7.69%
No answer	0	0.00%

Field summary for AIDA(3a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [It was easy to understand the code.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	23.08%
weakly disagree (A3)	3	23.08%
undecided (A4)	0	0.00%
weakly agree (A5)	5	38.46%
agree (A6)	2	15.38%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(3b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [It took much effort to understand the code.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	3	23.08%
undecided (A4)	2	15.38%
weakly agree (A5)	5	38.46%
agree (A6)	3	23.08%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(4a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [It seems that there are no unused instructions.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	1	7.69%
undecided (A4)	5	38.46%
weakly agree (A5)	3	23.08%
agree (A6)	2	15.38%
strongly agree (A7)	2	15.38%
No answer	0	0.00%

Field summary for AIDA(4b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are too much unused instructions.]

Answer	Count	Percentage
strongly disagree (A1)	1	7.69%
disagree (A2)	3	23.08%
weakly disagree (A3)	6	46.15%
undecided (A4)	2	15.38%
weakly agree (A5)	1	7.69%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(5a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [I think the code is correctly decompiled.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	1	7.69%
undecided (A4)	5	38.46%
weakly agree (A5)	5	38.46%
agree (A6)	2	15.38%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(5b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The decompiled code seems to be incorrect.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	23.08%
weakly disagree (A3)	6	46.15%
undecided (A4)	4	30.77%
weakly agree (A5)	0	0.00%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(6a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The conditions are too complex.]

Answer	Count	Percentage
strongly disagree (A1)	4	30.77%
disagree (A2)	2	15.38%
weakly disagree (A3)	4	30.77%
undecided (A4)	1	7.69%
weakly agree (A5)	1	7.69%
agree (A6)	1	7.69%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(6b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The code contains too many intermediate results.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	4	30.77%
weakly disagree (A3)	5	38.46%
undecided (A4)	1	7.69%
weakly agree (A5)	2	15.38%
agree (A6)	1	7.69%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(6c)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The line length is too long]

Answer	Count	Percentage
strongly disagree (A1)	4	30.77%
disagree (A2)	5	38.46%
weakly disagree (A3)	4	30.77%
undecided (A4)	0	0.00%
weakly agree (A5)	0	0.00%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(7a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [Many variables have useless copies.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	4	30.77%
weakly disagree (A3)	3	23.08%
undecided (A4)	2	15.38%
weakly agree (A5)	4	30.77%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(7b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are no useless copies of variables.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	23.08%
weakly disagree (A3)	5	38.46%
undecided (A4)	2	15.38%
weakly agree (A5)	2	15.38%
agree (A6)	1	7.69%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(8a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The variable types seem to be reasonable.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	7.69%
weakly disagree (A3)	2	15.38%
undecided (A4)	4	30.77%
weakly agree (A5)	4	30.77%
agree (A6)	2	15.38%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(8b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [Some variable types seem to be wrong.]

Answer	Count	Percentage
strongly disagree (A1)	1	7.69%
disagree (A2)	1	7.69%
weakly disagree (A3)	3	23.08%
undecided (A4)	4	30.77%
weakly agree (A5)	1	7.69%
agree (A6)	3	23.08%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(9a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are no variables that only store unnecessary intermediate constants.]

Answer	Count	Percentage
strongly disagree (A1)	2	15.38%
disagree (A2)	2	15.38%
weakly disagree (A3)	3	23.08%
undecided (A4)	2	15.38%
weakly agree (A5)	2	15.38%
agree (A6)	2	15.38%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDA(9b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are too many variables that just store intermediate constants.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	23.08%
weakly disagree (A3)	3	23.08%
undecided (A4)	4	30.77%
weakly agree (A5)	3	23.08%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDAQ1(SQ001)

What is the output of the function for the parameters [arg_1 = 0 and arg_2 = 1]

Answer	Count	Percentage
0 (A1)	0	0.00%
1 (A2)	0	0.00%
2 (A3)	11	84.62%
3 (A4)	2	15.38%
4 (A5)	0	0.00%
5 (A6)	0	0.00%
I do not know (A7)	0	0.00%
No answer	0	0.00%

Field summary for AIDAQ1(SQ002)

What is the output of the function for the parameters [arg_1 = 1 and arg_2 = 0]

Answer	Count	Percentage
0 (A1)	0	0.00%
1 (A2)	4	30.77%
2 (A3)	6	46.15%
3 (A4)	1	7.69%
4 (A5)	0	0.00%
5 (A6)	0	0.00%
I do not know (A7)	2	15.38%
No answer	0	0.00%

Field summary for AIDAQ2

Which computation needs more recursive calls?

Answer	Count	Percentage
arg_1 = 1 and arg_2 = 2 (A1)	5	38.46%
arg_1 = 2 and arg_2 = 1 (A2)	8	61.54%
No answer	0	0.00%

Field summary for AIDAQ4

What do you like or dislike about the above sample?

Answer	Count	Percentage
Answer	5	38.46%
No answer	8	61.54%

ID	Response
40	var_0 seems redundant arg_2 seems to have type unsigned_int and not int
79	Recursive calls
95	Like : It compiles, and therefore is easier to test dynamically for programmers. Doesn't use go-tos. Seems to be no dead code. Dislike : It is still hard to decipher what the function is attempting to compute.
96	from my observations, recursion appears rarely in real-world code.
109	Inconsistent datatypes. Mixing windows typedefs (DWORD) with default c types (unsigned int) and pseudo types (__int64) feels unclean. Personally I'd prefer only using the types from cstd.h (uint8_t, int64_t, ...), though int8_t vs. char is debatable.

Field summary for AGhidra(1a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The used control-flow structures are appropriate.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	0	0.00%
undecided (A4)	7	77.78%
weakly agree (A5)	2	22.22%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(1b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The control-flow is strangely restructured.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	11.11%
weakly disagree (A3)	2	22.22%
undecided (A4)	2	22.22%
weakly agree (A5)	2	22.22%
agree (A6)	2	22.22%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(3a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [It was easy to understand the code.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	1	11.11%
weakly disagree (A3)	5	55.56%
undecided (A4)	0	0.00%
weakly agree (A5)	2	22.22%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(3b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [It took much effort to understand the code.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	1	11.11%
undecided (A4)	1	11.11%
weakly agree (A5)	3	33.33%
agree (A6)	4	44.44%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(4a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [It seems that there are no unused instructions.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	11.11%
weakly disagree (A3)	1	11.11%
undecided (A4)	2	22.22%
weakly agree (A5)	2	22.22%
agree (A6)	2	22.22%
strongly agree (A7)	1	11.11%
No answer	0	0.00%

Field summary for AGhidra(4b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are too much unused instructions.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	1	11.11%
weakly disagree (A3)	2	22.22%
undecided (A4)	3	33.33%
weakly agree (A5)	0	0.00%
agree (A6)	2	22.22%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(5a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [I think the code is correctly decompiled.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	1	11.11%
weakly disagree (A3)	0	0.00%
undecided (A4)	2	22.22%
weakly agree (A5)	5	55.56%
agree (A6)	1	11.11%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(5b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The decompiled code seems to be incorrect.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	3	33.33%
undecided (A4)	5	55.56%
weakly agree (A5)	1	11.11%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(6a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The conditions are too complex.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	5	55.56%
weakly disagree (A3)	1	11.11%
undecided (A4)	2	22.22%
weakly agree (A5)	1	11.11%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(6b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The code contains too many intermediate results.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	1	11.11%
weakly disagree (A3)	1	11.11%
undecided (A4)	3	33.33%
weakly agree (A5)	2	22.22%
agree (A6)	1	11.11%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(6c)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The line length is too long]

Answer	Count	Percentage
strongly disagree (A1)	4	44.44%
disagree (A2)	2	22.22%
weakly disagree (A3)	3	33.33%
undecided (A4)	0	0.00%
weakly agree (A5)	0	0.00%
agree (A6)	0	0.00%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(7a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [Many variables have useless copies.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	2	22.22%
weakly disagree (A3)	2	22.22%
undecided (A4)	1	11.11%
weakly agree (A5)	2	22.22%
agree (A6)	0	0.00%
strongly agree (A7)	1	11.11%
No answer	0	0.00%

Field summary for AGhidra(7b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are no useless copies of variables.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	2	22.22%
weakly disagree (A3)	1	11.11%
undecided (A4)	1	11.11%
weakly agree (A5)	2	22.22%
agree (A6)	1	11.11%
strongly agree (A7)	1	11.11%
No answer	0	0.00%

Field summary for AGhidra(8a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [The variable types seem to be reasonable.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	0	0.00%
weakly disagree (A3)	0	0.00%
undecided (A4)	1	11.11%
weakly agree (A5)	5	55.56%
agree (A6)	3	33.33%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(8b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [Some variable types seem to be wrong.]

Answer	Count	Percentage
strongly disagree (A1)	0	0.00%
disagree (A2)	3	33.33%
weakly disagree (A3)	0	0.00%
undecided (A4)	4	44.44%
weakly agree (A5)	1	11.11%
agree (A6)	1	11.11%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidra(9a)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are no variables that only store unnecessary intermediate constants.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	0	0.00%
weakly disagree (A3)	2	22.22%
undecided (A4)	1	11.11%
weakly agree (A5)	2	22.22%
agree (A6)	2	22.22%
strongly agree (A7)	1	11.11%
No answer	0	0.00%

Field summary for AGhidra(9b)

Please consider the following decompiled function and answer the questions below. Feel free to use the editor as you would normally do (e.g. by renaming variables). [There are too many variables that just store intermediate constants.]

Answer	Count	Percentage
strongly disagree (A1)	1	11.11%
disagree (A2)	2	22.22%
weakly disagree (A3)	0	0.00%
undecided (A4)	0	0.00%
weakly agree (A5)	3	33.33%
agree (A6)	3	33.33%
strongly agree (A7)	0	0.00%
No answer	0	0.00%

Field summary for AGhidraQ1(SQ001)

What is the output of the function for the parameters [arg_1 = 0 and arg_2 = 1]

Answer	Count	Percentage
0 (A1)	0	0.00%
1 (A2)	0	0.00%
2 (A3)	7	77.78%
3 (A4)	0	0.00%
4 (A5)	1	11.11%
5 (A6)	0	0.00%
I do not know (A7)	1	11.11%
No answer	0	0.00%

Field summary for AGhidraQ1(SQ002)

What is the output of the function for the parameters [arg_1 = 1 and arg_2 = 0]

Answer	Count	Percentage
0 (A1)	1	11.11%
1 (A2)	0	0.00%
2 (A3)	5	55.56%
3 (A4)	0	0.00%
4 (A5)	0	0.00%
5 (A6)	0	0.00%
I do not know (A7)	3	33.33%
No answer	0	0.00%

Field summary for AGhidraQ2

Which computation needs more recursive calls?

Answer	Count	Percentage
arg_1 = 1 and arg_2 = 2 (A1)	3	33.33%
arg_1 = 2 and arg_2 = 1 (A2)	6	66.67%
No answer	0	0.00%

Field summary for AGhidraQ4

What do you like or dislike about the above sample?

Answer	Count	Percentage
Answer	4	44.44%
No answer	5	55.56%

ID	Response
58	the variables name are too similar and have to manually change them, if not very confusing.
68	dislike: too many type casts
114	Dislike: var_1 = arg_1 and var_2 = arg_2 and vice versa appears to only add to the confusion.
116	Dislike: arguments are modified, making it hard to keep track of their values Like: Short statements

Field summary for BQ1Q1 [1]

Please consider the following three samples and order them from your favourite to least favourite:
Sample 1 Sample 2 Sample 3 [Ranking 1]

Answer	Count	Percentage
Sample 1 (A1)	28	75.68%
Sample 2 (A2)	6	16.22%
Sample 3 (A3)	3	8.11%

Field summary for BQ1Q1 [2]

Please consider the following three samples and order them from your favourite to least favourite:
Sample 1 Sample 2 Sample 3 [Ranking 2]

Answer	Count	Percentage
Sample 1 (A1)	4	10.81%
Sample 2 (A2)	26	70.27%
Sample 3 (A3)	7	18.92%

Field summary for BQ1Q1 [3]

Please consider the following three samples and order them from your favourite to least favourite:
Sample 1 Sample 2 Sample 3 [Ranking 3]

Answer	Count	Percentage
Sample 1 (A1)	5	13.51%
Sample 2 (A2)	5	13.51%
Sample 3 (A3)	27	72.97%

Field summary for BQ1Q2

Do you prefer switch or if-else when there are at most 2 or 3 cases?

Answer	Count	Percentage
switch (A1)	23	62.16%
if-else (A2)	14	37.84%
Comments	12	32.43%
No answer	0	0.00%

ID	Response
37	Actually, I don't care in this case, but when writing code, I'd probably use if-else. So, this would be the natural choice here.
40	On fall-through cases (like in the example), I prefer switch. Else I prefer if-else on at most 3 cases.
43	Depends. If every switch with code has a break, it's fine, but not like this.
49	But that also depends on the complexity of the conditions.
61	This depends on the actual cases. For ranged cases, I prefer if-else. For a fixed set of cases (and an "else" / default case), I prefer switch.
70	Easier to analyse the control flow since we don't have to evaluate the conditions
80	Switch is easier to understand and follow the program flow.
81	no real pref., although switch is more readable
93	depends on complexity. if-else for simple ones, please.
96	in the given example, switch is more straight forward to read (even with the fall-through).
97	Dependent on the variable-type and if ranges are involved in the cases.
116	I usually prefer if-else for 2 or 3 cases, but do not prefer many nested ifs like in the sample above.

Field summary for BQ1Q3

If the value of a variable is known due to a condition, i.e., Branch or Switch, should it be propagated?

Answer	Count	Percentage
yes, the value should be propagated (A1)	25	67.57%
no, the value should not be propagated (A2)	12	32.43%
Comments	7	18.92%
No answer	0	0.00%

ID	Response
40	The variable name sometimes hold semantic information that would get lost if only the value gets propagated (like in sample 2, line 11) Even the used operation (like adding one) can contain useful meaning, even if the result is known (to be 6 like in the samples). Best case would be writing known values as comments behind statements where the value is known.
43	Without a hint, it's confusing. "Why are we setting the variable to 6 here?"
49	While this should be done in general, in some cases it would be helpful to leave the conditions as they are, to understand the underlying structure. Otherwise there would pop up some random numbers and pruned conditions and it is not clear what they mean.
61	This obscures future changes in the code as well as the intention of the assigned value, e.g., in line 16, the value 6 cannot be reconstructed as var_0 + 1 but appears as a constant.
65	It depends. Best would be per default no propagation but the end user should decide
70	Sometimes we can understand the semantics better when it is known how values are calculated
116	Operations should be preserved, e.g. the incrementing action in "var_0 + 1" may be more meaningful than the value being 6.

Field summary for BQ1Q4

What did you like most about your favourite sample?

Answer	Count	Percentage
Answer	33	89.19%
No answer	4	10.81%

ID	Response
37	It look very clean and concise. Except for the missing break in case 5 (missing breaks are always confusing), it's possible to immediately see what's going on.
40	Sample 1 - very easy to follow the control flow
43	no goto, no code in switch statements without break
49	clear structure
51	Clean
55	easy to understand the control flow conditions
58	simple and can be easily understood with single look
61	No gotos, cases are cleary distinguishable (no fall-through), return as soon as the result is known.
65	No
68	looks clean
69	Easiest to understand
70	Easy to analyse control flow since we just have to compare values with the cases in the switch
75	easy to read, but ties with switch (sample 1)
77	clear presentation of codes
78	Fewer condition checking and code is easy to follow
79	Easier to understand
80	Using case easy to follow and understand especially when the code is long
81	clear conditions to determine var0
82	Easy to read
83	neat
89	Cleaner & easier to understand
93	the complete switch-case.
94	clear structure
95	There were no lines where more than one condition had to be kept in mind.
96	the switch allows to read the code more or less linearly without the need to potentially re-read above conditions again.
97	More instantaneous overview of the cases for which additional instructions will be applied.
103	they are not very different actually, its a matter of syntax familiarity
107	Sample 3 is easier to interprete
108	if-else cases
109	switch
114	Clean hierarchy of codes
116	Very flat (not much nesting), no gotos
120	Switch-Case is super clean

Field summary for BQ1Q5

What did you dislike about your least favourite sample?

Answer	Count	Percentage
Answer	31	83.78%
No answer	6	16.22%

ID	Response
37	I think both, sample 2 and 3, are very cluttered. For sample 3: ifs combined with gotos? Please, no! However, I like that sample 3 has *2 instead of shifts.
40	Sample 2 - mixing of == and != as if-else conditions thus one has to be alert to not mix them up when following the control flow.
43	goto
49	deeper nested structure and gotos
51	Too complicated
55	messy, use of gotos makes code more difficult to follow, conditions are harder to understand
58	need longer time to understand the code
61	Goto, even worse than a fall-through.
65	the return statement is only inoked in the end
68	too many lines
69	prefer if else to switch
70	Messy control flow due to goto instruction
75	goto
77	too many nested statements
78	Code is hardest to follow
79	labels
80	The use of "!=" can mean there are a lot of possibilities in the variable as compared to the use of "=="
82	Harder to read
83	messy
89	goto label could potentially create some confusion in a bigger sample
93	lengthy unreadable code
94	goto and labels
95	The use of go-tos.
96	the negative condition in the first if feels less natural than in the other one (sample 3).
97	Too many if-indent-blocks, the goto instruction; it made the overall control-flow less obvious.
103	they are not very different actually, its a matter of syntax familiarity
108	Label
109	GOTO!
114	Labels add to the confusion.
116	Has nested ifs and has a goto
120	Its long and confusing. One has to keep to many things in mind

Field summary for BQ2Q1 [1]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 1]

Answer	Count	Percentage
Sample 1 (A1)	3	8.11%
Sample 2 (A2)	4	10.81%
Sample 3 (A3)	30	81.08%

Field summary for BQ2Q1 [2]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 2]

Answer	Count	Percentage
Sample 1 (A1)	28	75.68%
Sample 2 (A2)	3	8.11%
Sample 3 (A3)	6	16.22%

Field summary for BQ2Q1 [3]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 3]

Answer	Count	Percentage
Sample 1 (A1)	6	16.22%
Sample 2 (A2)	30	81.08%
Sample 3 (A3)	1	2.70%

Field summary for BQ2Q2

Do you prefer switch or if-else in a switch-case?

Answer	Count	Percentage
switch (A1)	6	16.22%
if-else (A2)	16	43.24%
if-else if switch has at most 2 cases and switch otherwise (A3)	15	40.54%
Comments	10	27.03%
No answer	0	0.00%

ID	Response
37	Actually, none of the above. In my opinion, the cases of a switch stmt. should be rather short and not contain any other control flow constructs.
43	depends on the context
61	Depends on the situation (ranges vs fixed numbers). Here, if-else seems to be advantageous.
80	If-else is also good if there are not too many if-else inside a if-else
81	if conditions are simple switch conditions, switch is alright, however if conditions are complex, i am more used to "parsing" if-else structure as you will need to mentally interpret the switch as if-else.
83	Actually, if there is a 4th choice where there are only 2 cases, but the switch condition is complex but actually only evaluates to either true or false (like in sample 2), then if-else is better.
96	for comparison of value ranges (0), opposite to explicit values like in the previous example, switch feels less natural and harder to comprehend (bitmask/shift magic is also harder to understand for beginners).
97	I tend to prefer if-else for ranges, except if the boundaries are few, e.g., 2, and the switch can be well-adapted to these boundaries.
109	As before, in general I prefer switch, but in this case it cannot properly be represented as switch. Representing it as has been done here is ridiculous. Don't do switch, if *in*equalities are involved.
120	The code seems less crowded with switch

Field summary for BQ2Q3

Do you prefer "else if" as one expression, if possible, (as in sample 3 lines 10 and 14) or separated into two expressions (as in sample 1 lines 16/17)?

Answer	Count	Percentage
one expression (A1)	34	91.89%
separated expressions (A2)	3	8.11%
Comments	5	13.51%
No answer	0	0.00%

ID	Response
70	Easier to understand. If the "else if" is separated into two expressions, we will still annotate them as one expression during analysis.
81	visually easier to determine if-else block
83	even better if can reduce from "if(A) {} else if(!A) {}" to simply "if(A) {} else {}"
89	One expression makes it cleaner, easier to encapsulate the logic
97	Separated solely if it makes sense to include multiple separations again, e.g., by separating the positive case into 5.

Field summary for BQ2Q4

What did you like most about your favourite sample?

Answer	Count	Percentage
Answer	32	86.49%
No answer	5	13.51%

ID	Response
37	Again, it's short and concise. No deeply nested control flow structures, clean conditions.
40	Easy to follow control flow
43	it's not my least favourite sample
49	Easy to read, only one condition dazzled me
51	Clean
55	very clear and concise control flow conditions, compared to the other samples.
58	more compact and easily understandable
61	Without studying bitshifts or triple nested if statements, the purpose of the function became clear within a second.
65	That the format string is directly resolved instead seeing the address of the format string
68	short
69	easiest to understand
70	Simple control flow
75	straight forward inequality checks
77	concise way of presentation
78	Easy to follow and conditional checks were easy to understand
79	Easier to understand
80	Short and clear
81	short and neat, making analysing conditions easy
82	Easier to read.
83	less nested-if; no redundant nested-if with same condition
89	Clean and directly easy to see the control flow & conditions
93	nice & short please
94	if-else was easy to follow
95	Seems closest to human-written source-code
96	among the nesting depth of the given examples and concise logical split of value ranges captured by the ifs
97	Less unnecessary instructions.
103	data type conversion is not required
108	comprehensible flow
109	The format string is visible.
114	neat. straightforward.
116	"else if" is one expression when there is only one block after else
120	The more python-esque indentation. C/C++ style indentation is ugly in my opinion

Field summary for BQ2Q5

What did you dislike about your least favourite sample?

Answer	Count	Percentage
Answer	31	83.78%
No answer	6	16.22%

ID	Response
37	Switch in switch is awful!
40	Noise from typecasts and switch statements, where if-else would have been more natural.
43	so many fucking casts
49	unnecessary if-block where the reason is not clear
51	Too complicated
55	nested switch case, conditions that require more time to understand (even if it may be more representative of the asm instructions)
58	switch case in this scenario become rather confusing.
61	I usually don't like nested switches. The switch head is hard to read and understand.
65	the address of the format string printer and using switch case for this limeted "cases"
68	long
69	more complicated than the other 2 samples
70	Unnecessary conditions like the nested if
75	insane switch statements
78	Complex conditional check
79	multiple if
80	The use of hex number and & operation is more difficult to understand
81	if all the 3 samples are the same, unnecessary opertions
82	Can be confusing at times.
83	switch statement makes it not obvious only 2 cases possible i.e. it is an "if-else"
89	Bitshifts will occur more mental effort to reverse and understand the same thing
93	lengthy and harcoded numbers when unneeded
94	switch in switch is confusing
95	Use of casting and bit-level operations.
96	bitmask/shift magic requires more intrinsic knowledge about number representation than straightforward if/else constructs.
97	Outer switch instructions feels unnecessary complicated to read.
103	data type conversion is required
108	switch cases
109	The switch obscures the semantics!
114	lots of logical thinking needed.
116	Unnecessary extra pointer variable (no use of array)
120	Too many ifs mess with ones head, because you have to keep a lot of stuff in mind

Field summary for BQ3Q1 [1]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 1]

Answer	Count	Percentage
Sample 1 (A1)	34	91.89%
Sample 2 (A2)	2	5.41%
Sample 3 (A3)	1	2.70%

Field summary for BQ3Q1 [2]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 2]

Answer	Count	Percentage
Sample 1 (A1)	3	8.11%
Sample 2 (A2)	34	91.89%
Sample 3 (A3)	0	0.00%

Field summary for BQ3Q1 [3]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 3]

Answer	Count	Percentage
Sample 1 (A1)	0	0.00%
Sample 2 (A2)	1	2.70%
Sample 3 (A3)	36	97.30%

Field summary for BQ3Q2

Do you think goto is a good choice when breaking out of the inner loop?

Answer	Count	Percentage
yes, goto is a good choice (A1)	14	37.84%
no, goto should not be used (A2)	23	62.16%
Comments	18	48.65%
No answer	0	0.00%

ID	Response
40	As long as goto is only used similar to a "break;" statement, it can help readability. But if mixed with other control flow constructs it often reduces readability.
43	As you see in Sample 1, it's just unnecessary
49	In this particular case, the goto is unnecessary, because the function can terminate. But in general a goto to jump out of an inner loop can improve the readability.
55	If we are only comparing between sample 2 and 3, then yes it makes the decompiled code easier to read.
61	Just do the printf and return statement there.
69	easy to understand
70	Having a goto is more intuitive to understand
75	i don't write C using goto, therefore it should not be used in decompilation
78	prefer using break
80	Prefer break than goto statement
81	if the goto does not lead to another jump, goto is easier to read
93	goto is bad, but still better then lengthy code!
95	In this case there was only one goto and it acted like a finally block, and so was easy enough to understand and did help in some ways, too. Where you begin to have multiple gotos and that don't necessarily return, it may become harder to understand. I think the analyst can get used to the goto construct eventually, and it does help with reducing code repetition in some cases, but it adds to the learning curve as it's not something normally used by those coming from a programming background.
96	It feels like the redundancy of the "return printf()" does not hurt here. Because it's the last statement within the loop(s), it allows directly to "stop" trying to understand the context.
97	Goto is fine if there are at most two different labels to jump to and if the switch-body is of small-ish size (probably
109	Gotos obscure the control flow, make it harder to read. Also feels like the decompiler was just lazy.
116	It is fine here when used singly, but in general multiple gotos and labels make code harder to read.
120	It messes with my head

Field summary for BQ3Q3

Do you think it is a good idea to duplicate (return-)statements to avoid complicated conditions or gotos?

Answer	Count	Percentage
copy statements to simplify the structure (A1)	30	81.08%
never copy a statement (A2)	7	18.92%
Comments	16	43.24%
No answer	0	0.00%

ID	Response
40	If a statement is complicated, you may end up trying to understand it twice instead of once. I also sometimes accidentally overlook return statements in the code when skimming over it. Just one return at the end decreases the risk of that happening.
49	In general duplication of code is bad and yields convoluted code, but to a certain extend and to simplify the structure, this might help.
55	In this case, it produced much cleaner decompiled code, so yes.
61	If it's a single (or reasonable short) statement: copy Otherwise: Extract as a new method?
65	for me it makes the analysing a little bit faster
70	Complicated branch structures are harder to trace
78	easier to understand the code
80	Too many return statement complicate the code
89	Depends, copying could make it simplified in larger functions
93	don't do that if you can avoid it.
95	Again, this more similar to how a programmer would do it, I think.
96	yes, see above.
97	Copy statements if restructuring/ rewriting them together afterwards is mostly easily possible.
109	I don't see any downsides to copying statements. In particular if it results in cleaner code, i.e. no goto.
116	I think this is useful if the destination statement(s) are simple and short. Perhaps it can be combined with some indicator to tell the reader this statement is duplicated (and not a new code block).
120	Goto is super ugly. I want decompilation to be easy to understand and not adhere to clean code rules

Field summary for BQ3Q4

What did you like most about your favourite sample?

Answer	Count	Percentage
Answer	30	81.08%
No answer	7	18.92%

ID	Response
37	Same as always: it's the shortest one and it doesn't contain any unusual constructs.
40	Easy to follow control flow, the scanf-string was inserted in the code.
43	It's short
49	short, intention of the function is very clear
51	Simple
55	Much easier to read, looks more like code someone will write.
58	compact and easy to understand
61	No gotos, easy to understand, for loop without a break used which further simplifies the method.
65	format string + the simplicity
68	simple and short
69	easy to understand and straightforward
70	Easy to see when loops break or function returns
75	looks clean
78	Least number of lines and easy to understand
79	short
80	Short and clear
81	short and easier to read loops
82	It copied the return statements so it is easier to understand.
83	simplest structure among three. even better if can use break
89	Easy to see control flow
93	nice & short, no gotos.
94	short and readable
95	It looked closest to something I'd program.
96	linear readability.
97	Conciseness, by far most obvious control flow & semantics.
108	for loop
109	Clean, sematics are clear on first pass.
114	compact, neat, nicely structured
116	The for-loop was recognised, making it simpler to read.
120	Its clean and easy to read

Field summary for BQ3Q5

What did you dislike about your least favourite sample?

Answer	Count	Percentage
Answer	30	81.08%
No answer	7	18.92%

ID	Response
37	Deep nesting, lots of variables; this looks totally cluttered.
40	Too many local variables
43	It's so fucking long and uses a bunch of variables
49	too many variables to keep track of
51	Complicated
55	Too many variables to track.
58	long length of code
61	Useless variables, too long, while(true) where not necessary.
65	too many variables and too complicated for this kind of "task"
68	too long
69	long and complicated
70	Complicated control flow
75	looks bloated
78	To many variables and longest of among the 3
79	long lines of code
80	Too many variable and operation and the use of !=
81	unnecessary complexity, can be made simplier
82	Nested while loops
83	complex
89	Too many branches & conditions
93	too lengthy.
94	confusing structure
95	It seemed most complex.
96	cluttered, too many variables compared to the others.
97	Way too many unnecessary case-changes, variables; didn't use common counter-variable names, as the 'i' in sample 1. I much prefer the x += y statement over x = x + y. Too many breaks in combination with the 'true' while-condition.
108	while True statement
109	The semantics are completely unclear. In particular the while(1) with break should be avoided if possible.
114	I dislike my least favourite sample because I have my reasons. Some of the reasons are as follow: it is too verbosive with quite a few variables involved. Another such reason is that the additional variables and lines more difficult to read. Following that, the variables, lines and nesting of the conditional variables added to the difficulty.
116	Long while loop with multiple breaks.
120	Its long, its ugly and it have to keep too much stuff in mind

Field summary for BQ4Q1 [1]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 1]

Answer	Count	Percentage
Sample 1 (A1)	21	56.76%
Sample 2 (A2)	16	43.24%
Sample 3 (A3)	0	0.00%

Field summary for BQ4Q1 [2]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 2]

Answer	Count	Percentage
Sample 1 (A1)	16	43.24%
Sample 2 (A2)	20	54.05%
Sample 3 (A3)	1	2.70%

Field summary for BQ4Q1 [3]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 3]

Answer	Count	Percentage
Sample 1 (A1)	0	0.00%
Sample 2 (A2)	1	2.70%
Sample 3 (A3)	36	97.30%

Field summary for BQ4Q2

Do you prefer while(true){} or do{}while(true)?

Answer	Count	Percentage
while (A1)	37	100.00%
do-while (A2)	0	0.00%
Comments	10	27.03%
No answer	0	0.00%

ID	Response
43	Depends on the situation, eh?
61	Especially for longer loop-bodies as we see in the examples, while eases the readability. As we only talk about while(true), there is absolutely no reason for dowhile(true) imho
69	more logical flow
70	Can see the condition at once
81	no actual pref. just that if there is a nested while or nested dowhile, it has to be the same (ie, all loops using dowhile or all loops using while)
93	definitely while
96	while shows the condition under which execution happens, do-while appears to require an additional if to branch out in case the condition is violated -> feels a bit cluttered.
97	Exceptions are cases for which only one (possibly longer) break-condition exists that can be nicely caught before the main while-body. In case loops are nested, I strongly prefer while over do-while.
109	When coding I *never* use do-while. I'd rather unroll the loop once. I find it easier to read.
120	do-while leads to crowded code

Field summary for BQ4Q3

Do you prefer continue-statements and only one if-statement or no continue-statement and an if-else-statement?

Answer	Count	Percentage
continue and if (A1)	15	40.54%
if-else (A2)	22	59.46%
Comments	12	32.43%
No answer	0	0.00%

ID	Response
37	But I think this depend on the sample at hand. Might be different for different samples.
40	Everything that reduces perceived code complexity helps.
43	Also depends on the situation
49	But only if the body of the if-statement is short and at the top of the loop
61	Depends on how long the remaining code (after the continue) is. Continue as a "jump" is not always nice, but it clearly indicates that I do not have to care about the rest of the body in this iteration. When the else part only spans 3 or 4 lines, I would rate if-else as reasonable as well.
69	easier to understand
75	hard choice, but continue seems a bit easier to read
81	it is easier to follow an if-else block, as the continue-if condition introduces another mental note of a condition
96	in this example, if-else felt like it achieves better logical segmentation of the while's body.
97	If the if-condition and body is short, while the else is long (and the corresponding negated if-condition would be worse to read), continue+if can be better.
109	When coding I like to omit the else in favour of a continue. Thus I prefer reading code that was written that way.
120	if-else leads to crowded code

Field summary for BQ4Q4

What did you like most about your favourite sample?

Answer	Count	Percentage
Answer	25	67.57%
No answer	12	32.43%

ID	Response
37	Hard to explain in this case. It just looks more "natural". Maybe bacause we have one indent-level less.
43	an outer for loop
49	the two while-loops made the scope of the values clear, decrement and increment operations
51	Clean
55	the exit condition variables are more easy to track compared to the rest.
58	the loop is easier to understand in sample 1
61	for-loops, less nesting than the other samples, one can easily follow the flow
65	Using continue and that we have a for and a while loop
68	prefer having for loop
69	easiest to understand amongst the 3
75	for loop
77	involves less conditional statements
78	Easiest to follow and I did not have to deal with pointers
79	short
80	While and if is easier to understand compared to for followed by while or do (while true)
81	structurally easier to visualise the code
82	Easier to follow the flow.
93	better structuring, a bit shorter and nicer.
95	The loop constructs were simplest.
96	inc/dec operators.
97	Clearer control-flow, especially in regards to which & how many variables that are part of a control-flow-condition are changed inside the body.
108	Structured view
109	Fairly clean. For loop.
116	Did not have too much redirection that breaks the reading flow (goto, break, continue)
120	No else branches

Field summary for BQ4Q5

What did you dislike about your least favourite sample?

Answer	Count	Percentage
Answer	25	67.57%
No answer	12	32.43%

ID	Response
37	This looks obscure somehow.
40	while(true) and goto inside a loop
43	Why are we using goto again
49	logic is not clear at all, if one has not seen the two other samples beforehand
51	Complicated
55	harder to keep track of when/how the loop exits
58	dislike 'goto' and 'while loop' being used together as need to jump around the code more often.
61	Mix of while and do while, goto, nesting-levels
65	goto statements
68	hate do-while
69	do while loop
75	goto
78	Most difficult to follow
79	long lines of code
80	too many combination, do while true , goto and break
81	mixture of dowhile and while
93	they were all relative bad, 2 & 3.
94	goto
95	Use of goto to jump to part of an outer loop
96	compared to the other two, it's harder to follow what is happening why (loops, conditions)
97	long, nested do-while loops and the label being defined inside the while loop
108	Label
109	do-while and goto!
116	it mixes while with do-while, and it's difficult to keep track of conditions above and below
120	Long and crowded

Field summary for BQ5Q1 [1]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 1]

Answer	Count	Percentage
Sample 1 (A1)	20	54.05%
Sample 2 (A2)	15	40.54%
Sample 3 (A3)	2	5.41%

Field summary for BQ5Q1 [2]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 2]

Answer	Count	Percentage
Sample 1 (A1)	13	35.14%
Sample 2 (A2)	19	51.35%
Sample 3 (A3)	5	13.51%

Field summary for BQ5Q1 [3]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 3]

Answer	Count	Percentage
Sample 1 (A1)	4	10.81%
Sample 2 (A2)	3	8.11%
Sample 3 (A3)	30	81.08%

Field summary for BQ5Q3

Do you prefer the condition if(var == 0) or if(!var)?

Answer	Count	Percentage
if(var == 0) (A1)	30	81.08%
if(!var) (A2)	7	18.92%
Comments	12	32.43%
No answer	0	0.00%

ID	Response
37	If you want to match the number 0 use 0. Use ! only for booleans or "wrong things".
43	This is clearly not Python
61	As we talk about an int here, I like comparison to another int better
68	doesn't really matter, but would prefer memcmp/strcmp == 0 instead of !memcmp/strcmp
70	var==0 seems more explicit or more intuitive to understand than !var
81	condition is simpler to understand
93	in general (!var). but that really doesn't make a big difference.
95	In this case it looks like it is meant to be a boolean, so !var makes sense.
97	Assuming that the knowledge of var being int-castable is found nearby/ easily remembered
109	I much prefer explicit zero checks "== 0", takes load off my brain. With "if(var)" I always have to take a moment to remind myself what that means.
114	Undecided. Depends on the code block following. Sometimes it is more natural for a if (!var).
116	I don't feel strongly about this one. ! is convenient but could be misleading if the variable is used as a counter and not a flag.

Field summary for BQ5Q4

What did you like most about your favourite sample?

Answer	Count	Percentage
Answer	24	64.86%
No answer	13	35.14%

ID	Response
37	The control flow structure looks best to me.
40	string argument for scanf inserted into the code.
43	no strange auxiliary variables for the scanf
49	clear data and pointer types
51	Cleaner
55	least disliked
58	more simplified
61	i and j as for loop variables ease the understanding as the names clearly show their purpose.
65	despite var_ also increment vars like i and j are used and the == 0
68	short and clean
75	variable naming looks clean and it has for loops
78	Easiest to follow
79	short
80	simple variable
81	for loops and conditions are clear
93	a bit better structured than the others.
94	good and easy variable names
95	Seemed to recover semantics best, e.g. boolean in line 21 and arithemetic in line 23
96	"cleanest" for loops
97	for loop with ++ statements, no totally-unused variables
108	if == 0 statement
109	Better local variable detection (no unnecessary arrays) and explicit zero checks.
116	The for-loops are elegant and easy to read.
120	Shorter and for-loop

Field summary for BQ5Q5

What did you dislike about your least favourite sample?

Answer	Count	Percentage
Answer	21	56.76%
No answer	16	43.24%

ID	Response
37	Here we have the worst structure of the three samples.
43	no for loops
49	unneeded variable, variable initialization in the while-condition
51	Complicated
55	while loop when for loop works
58	dislike the while-loop in this scenario, prefer for-loop in other sample
61	undefined4 var_2; Many variables, while loops, typo in "chosed"
65	undefined4 and the while while nesting
68	uses while instead of for loop
75	while (var_3 = 0, var_5 < var_0)
78	Hardest to follow and pointers
79	long lines of code
80	mixture of while and if-else
81	unnecessary complexity; undefined?
93	undefined type is a no go.
95	While loops used instead of for loops.
96	I think for loops are easier to read than while loops.
97	The x = x+y statements, especially since they are just incremented in such a manner much faster understood by a for-loop.
109	While instead of for
116	Used while loops when for-loops were possible, but this was not too bad.
120	I hate while-loops

Field summary for BQ6Q1 [1]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 1]

Answer	Count	Percentage
Sample 1 (A1)	0	0.00%
Sample 2 (A2)	36	97.30%
Sample 3 (A3)	1	2.70%

Field summary for BQ6Q1 [2]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 2]

Answer	Count	Percentage
Sample 1 (A1)	11	29.73%
Sample 2 (A2)	1	2.70%
Sample 3 (A3)	25	67.57%

Field summary for BQ6Q1 [3]

Please consider the following three samples and order them from your favourite to least favourite: .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3 [Ranking 3]

Answer	Count	Percentage
Sample 1 (A1)	26	70.27%
Sample 2 (A2)	0	0.00%
Sample 3 (A3)	11	29.73%

Field summary for BQ6Q2

Do you prefer for-loops or while-loops?

Answer	Count	Percentage
for-loops (A1)	33	89.19%
while-loops (A2)	4	10.81%
Comments	14	37.84%
No answer	0	0.00%

ID	Response
37	In my opinion for loops are always preferable as long as you keep any magic or complex things out of the loop header. Don't do any complex or unexpected computations in your loop header.
43	It's easier to see how often they loop
49	depends, but in this case the for-loops are better
61	Loop behavior is much more clear. Always use for loop if you iterate over a fixed range. Even when you have to use break.
69	for loops are neater and shorter
70	Can more explicitly see how conditions are calculated
81	easier to visualise the condition
93	for-loops, maybe. not a big difference. I probably don't care much.
95	When it actually makes sense (not e.g. for(; condition ;))
96	conditions in the for loops at beginning of their body makes it easy to understand how they'll work (even with the break).
97	In case the loop-condition variables are changed in more complex ways, a while-loop with a clear break-condition is better.
109	The semantics are more easy to decipher.
114	No preference.
120	while-loops make me sad. The loop body is more crowded

Field summary for BQ6Q4

What did you like most about your favourite sample?

Answer	Count	Percentage
Answer	24	64.86%
No answer	13	35.14%

ID	Response
37	It was the only one where I could understand what's going on whithin the first 5-10 seconds.
40	concise
43	It's short
49	clear logic, function call in the if-condition avoids unnecessary variable assignment
51	Cleaner
55	no wasted variables, easy to follow code
58	for-loop is easier to follow and do the dividable calculation.
61	For loops, variable namings (i,j), short, easy to follow
65	no unnecessary copy of vars
68	short and uses for loop
75	two clean looking for loops and division check in if condition
77	clear understanding of the codes
78	Easiest to follow and had the fewest line of code
80	short and easy to understand code structure
81	no while loops
93	nice & short, even nice variable names, that's unusual, and much better than the other two,
94	short and clear
95	It was most concise and the functionality was easy to figure out.
96	clean and straight forward
97	Very clear loop-behaviour from the first look on it.
108	for statement
109	It's pretty much the same as I would write the code. Two for loops, rather than nasty whiles.
116	For-loops are elegant and the counters are already named to defaults (i, j)
120	For-loops

Field summary for BQ6Q5

What did you dislike about your least favourite sample?

Answer	Count	Percentage
Answer	24	64.86%
No answer	13	35.14%

ID	Response
37	The while(true) loops were unexpected here.
40	do-while
43	The do-while loop
49	if-condition with break right after the function start distracts from the intention and should be a do-while-loop
51	Complicated
55	do while
58	the while loop make the code more confusing to do the dividable calculation
61	Return type, do while, does not follow the algorithms intuition.
65	copy vars for format string
68	hate do-whiles
75	do while loop
77	has several different conditional statements which may require longer time to understand the codes
78	Hardest to follow
80	Mixture of do, while and if-else make it hard to understand and trace
81	mix of dowihle and while
93	lengthy, ugly.
94	do{}
95	It used do-while loops and while loop where a for loop would have been more appropriate.
96	not a fan of the do-while like in the examples before
97	Combined use of var_3 as incrementing condition-variable and setting it as "return" value combined with break (l. 16)
108	do while
109	do {} while(true). Might as well read the disassembly rather than the decompiled code.
116	while-true with breaks
120	Do-while

Field summary for C1

We have decompiled the same function with different expression propagation limits. Please choose your favourite version with respect to the most optimal nesting complexity of instructions. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } limit 1limit 3limit 5limit 7limit 10

Answer	Count	Percentage
limit 1 (A1)	4	10.81%
limit 3 (A5)	7	18.92%
limit 5 (A2)	12	32.43%
limit 7 (A4)	4	10.81%
limit 10 (A3)	10	27.03%
Comments	13	35.14%
No answer	0	0.00%

ID	Response
37	I was balancing between 5 and 7. 1 and 3 have too many lines. In 7 I find the computation in the for loop header too complex, I also don't understand why it doesn't have var_4 = var_5. In 10 the lines are too complex. But that may be because of the arithmetic stuff. Lots of shifts and +- are more complex than calls to functions with meaningful names for instance. I. e. for arithmetic stuff I'd probably opt for shorter lines while I'd accept longer lines for function calls.
40	Limit 7 looks best here due to statement symmetry, but I assume that this does not generalize.
43	Only two operands per line in that long block of calculations please
49	7 and 10 are unnecessary complex, there are too many operations to keep track of 1 is useless 3 and 5 are better
51	Cleaner
61	Unsure between 5 and 3, but 5 is understandable without looking at each line for a minute or counting parenthesis.
69	least complex
75	seems like a sensible tradeoff between number of assingments and their complexity
81	fewer variables to track
93	ugh, but still limit 10 makes the overview better I think.
96	I think I like middleground (limit 5) best. What would be great though: To be able to decide this on a case-by-case, e.g. by having this available as a parameter in a UI.
107	Rest not familiar with the boolean function
120	It would never try and understand these without running the individual lines. Limit 10 makes it easiest for me to copy&paste

Field summary for C2

Please take a look at the following function and evaluate it with respect to long subexpressions that occur multiple times. Choose the most appropriate one. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } 234

Answer	Count	Percentage
2 (A1)	24	64.86%
3 (A2)	10	27.03%
4 (A3)	3	8.11%
Comments	7	18.92%
No answer	0	0.00%

ID	Response
43	I do not understand what you want from me here
49	4 is horrible, to much is happening at once and it is already difficult to keep track of opening and closing paranthesis updating var_3 in two steps seems reasonable
61	One more line does not hurt but eases readability.
81	less repetitive expressions
96	2 feels easiest to follow. But again, might be great to have such things available as a parameter.
116	It seems useful to replace long subexpressions with variables when the same subexpression is later used 2 or more times.
120	Again.. Too much calculating in head. So long lines for easy copy&paste

Field summary for C3

We have decompiled the same function with different configurations. Please choose up to three samples that are most comprehensible to you. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } Sample 1Sample 2Sample 3Sample 4Sample 5Sample 6

Sample 1	**13**	**35.14%**
Sample 2	16	43.24%
Sample 3	21	56.76%
Sample 4	18	48.65%
Sample 5	12	32.43%
Sample 6	6	16.22%

ID	**Response**
81	3rd choice. don't like to for loop condition

Field summary for CQ31

Please enter your comment here:

Answer	Count	Percentage
Answer	6	16.22%
No answer	31	83.78%

ID	Response
43	They all look the same to me
81	as above
93	all bad
95	In selected samples, it was easiest to identify key state variables and how they were changed across iterations. Sub-expression computations were not repeated unnecessarily, as in sample 6.
96	shorter lines might be easier to understand, allow to rename/comment more finely.
116	Anything but samples 1 and 6 is okay to read.

Field summary for C4

Please take a look at the following function that was decompiled either using a for-loop or using a while-loop. Choose the output that you prefer. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; }

Answer	Count	Percentage
for-loop (A1)	8	21.62%
while-loop (A2)	29	78.38%
Comments	15	40.54%
No answer	0	0.00%

ID	Response
37	As I mentioned earlier: in general for loops are better, but not if you have complex computations in the header. Also, this is a very unintuitive way to use a foor loop (empty, body, using the "counter" variable after the loop).
40	complicated expressions in loop header of for-loops is a no-go.
43	Why is that an empty for loop with a big condition, what is this nonsense
49	the assignment is much to long the resulting empty body seems weird also the assignment right after the body throws me of
61	More for layout than for the loop type.
70	For loop seems more complicated to understand when there are long expressions involved
75	I prefer easy to read conditions
80	while-loop for this code easier to understand compared to the for-loop
81	prefer the conditions for the loop to be simple
95	Prefer for key computation to be kept in the loop body rather than header.
96	while loop feels more appropriate because the loop body / loop variable update looks non-trivial (compared to increments etc).
97	Because the i = (...) statement in the for-loop is so long, a horizontal deviation as seen in the while-loop feels easier and clutters less with the long statements from the lines above.
109	In this case the for loop does not make it easier to comprehend the semantics. I feel like anything that does more updates that are more complex than *simple* arithmatics "i += " or "i *= " might be better off being a while loop. I would personally draw the line somewhere around "i = (i+) % ". That's the most complex statement I would still use a for loop for.
116	for-loop is more suitable when the update statement is short, and the inner block has content.
120	It does not mattert what I choose here.. This seems too complicated in its details and too simple in its interface. In a real world scenario I would not try to understand this and instead let it run with various inputs and extrapolate the functionality from that.

Field summary for C5 [1]

The output contains various of byte constant (from printable ASCII range) representation in the decompiler output. Rank them from most to least favourite. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } charchar-decimal-in-commentdecimaldecimal-char-in-comment [Ranking 1]

Answer	Count	Percentage
char (A1)	14	37.84%
char + decimal as comment (A2)	15	40.54%
decimal (A3)	1	2.70%
decimal + char as comment (A4)	7	18.92%

Field summary for C5 [2]

The output contains various of byte constant (from printable ASCII range) representation in the decompiler output. Rank them from most to least favourite. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } charchar-decimal-in-commentdecimaldecimal-char-in-comment [Ranking 2]

Answer	Count	Percentage
char (A1)	8	21.62%
char + decimal as comment (A2)	13	35.14%
decimal (A3)	4	10.81%
decimal + char as comment (A4)	12	32.43%

Field summary for C5 [3]

The output contains various of byte constant (from printable ASCII range) representation in the decompiler output. Rank them from most to least favourite. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } charchar-decimal-in-commentdecimaldecimal-char-in-comment [Ranking 3]

Answer	Count	Percentage
char (A1)	12	32.43%
char + decimal as comment (A2)	6	16.22%
decimal (A3)	6	16.22%
decimal + char as comment (A4)	13	35.14%

Field summary for C5 [4]

The output contains various of byte constant (from printable ASCII range) representation in the decompiler output. Rank them from most to least favourite. .tab { overflow: hidden; border: 1px solid #ccc; background-color: #f1f1f1; } /* Style the buttons that are used to open the tab content */ .tab button { background-color: inherit; color:black; float: left; border: none; outline: none; cursor: pointer; padding: 14px 16px; transition: 0.3s; } /* Change background color of buttons on hover */ .tab button:hover { background-color: #ddd; } /* Create an active/current tablink class */ .tab button.active { background-color: #ccc; } /* Style the tab content */ .tabcontent { display: none; padding: 6px 12px; border: 1px solid #ccc; border-top: none; } charchar-decimal-in-commentdecimaldecimal-char-in-comment [Ranking 4]

Answer	Count	Percentage
char (A1)	3	8.11%
char + decimal as comment (A2)	3	8.11%
decimal (A3)	26	70.27%
decimal + char as comment (A4)	5	13.51%

Field summary for CQ51

Please enter your comment here:

Answer	Count	Percentage
Answer	15	40.54%
No answer	22	59.46%

ID	Response
37	Maybe not directly related, but: if it's possible to detect that the code checks for capital letters e.g. then I'd prefer to have the condition altered to var >= 'A'.
43	Come on, either this is text, then I want to see a character, or it is a number, then I want to see a number, and if the decompiler is wrong, I want to easily see that by looking at what chars the numbers are
49	only the decimal is hard to understand without a ascii table right next to it...
61	As the method really works on chars and does not calculate with them, the single char is sufficient and the only relevant information for this method.
68	would be good if it's toggle-able
70	Would rather have a more simple view of char or decimal at one time and choose their representation manually (like in IDA)
80	Char make more sense than decimal for code the comment make the code hard to read (prefer mouse over to get the comments or comments at the end of line)
81	humans prefer reading what the condition means, but essential to know the decimal value
89	Could be unicode and i might not want it to display as ascii printable char at first. Would prefer decimal + char as comment
93	please hexadecimal instead of decimal. Who needs decimal anyway?
96	first choice should be concise (either dec or char), comments maybe as an optionally feature (via button).
97	The range-tests for characters (line 9) would be great as an individual if at the start.
109	In this particular case it is obvious (for a human) that it's actual characters rather than numbers, but I'd still like the decimal values in case the decompiler misinterpreted the use as characters.
116	if the byte variable is always compared to a printable-ascii value, char alone is great.
120	As many information as possible

Field summary for C6 [1]

Please consider the following variable naming schemes and rate them from favourite to least favourite: [Ranking 1]

Answer	Count	Percentage
Scheme 1 (var_0,...) (A1)	13	35.14%
Scheme 2 (v1,...) (A2)	12	32.43%
Scheme 3 (dog,...) (A3)	6	16.22%
Scheme 4 (a,...) (A4)	6	16.22%
Scheme 5 (loc_1,...) (A5)	0	0.00%

Field summary for C6 [2]

Please consider the following variable naming schemes and rate them from favourite to least favourite: [Ranking 2]

Answer	Count	Percentage
Scheme 1 (var_0,...) (A1)	11	29.73%
Scheme 2 (v1,...) (A2)	11	29.73%
Scheme 3 (dog,...) (A3)	4	10.81%
Scheme 4 (a,...) (A4)	1	2.70%
Scheme 5 (loc_1,...) (A5)	10	27.03%

Field summary for C6 [3]

Please consider the following variable naming schemes and rate them from favourite to least favourite: [Ranking 3]

Answer	Count	Percentage
Scheme 1 (var_0,...) (A1)	8	21.62%
Scheme 2 (v1,...) (A2)	10	27.03%
Scheme 3 (dog,...) (A3)	2	5.41%
Scheme 4 (a,...) (A4)	6	16.22%
Scheme 5 (loc_1,...) (A5)	11	29.73%

Field summary for C6 [4]

Please consider the following variable naming schemes and rate them from favourite to least favourite: [Ranking 4]

Answer	Count	Percentage
Scheme 1 (var_0,...) (A1)	2	5.41%
Scheme 2 (v1,...) (A2)	3	8.11%
Scheme 3 (dog,...) (A3)	9	24.32%
Scheme 4 (a,...) (A4)	15	40.54%
Scheme 5 (loc_1,...) (A5)	8	21.62%

Field summary for C6 [5]

Please consider the following variable naming schemes and rate them from favourite to least favourite: [Ranking 5]

Answer	Count	Percentage
Scheme 1 (var_0,...) (A1)	3	8.11%
Scheme 2 (v1,...) (A2)	1	2.70%
Scheme 3 (dog,...) (A3)	16	43.24%
Scheme 4 (a,...) (A4)	9	24.32%
Scheme 5 (loc_1,...) (A5)	8	21.62%

Field summary for C6F

Do you have any better ideas for variable naming schemes?

Answer	Count	Percentage
Answer	12	32.43%
No answer	25	67.57%

ID	Response
37	Sorry, unfortunately not. What would be important would be that function arguments are distinguishable from local variables. While I ranked the animal scheme rather low, I'm actually not sure if it would be a bad idea. Unconventionally, but short words might be harder to confuse and easier to remember than the short v-things. I would have to see a larger code sample with animals to evaluate this. Also, I'm undecided if it makes sense to name counter variables "counter_1" or something (can you could detect this?). Not sure if consistency would be preferable here.
43	Guess the function of the variable and pick a suitable name :3
49	No, but I don't like the naming schemes where only an integer is incremented in the scheme. This makes the variables visually hard to distinguish. When (or if) the exact meaning of the variable is clear, it can be renamed after all.
58	meaningful word that can help understand more about the variable data
61	dog, ... suggests a meaning of the name. Same does the usage of a,b,... (don't let that count up to i or j!). The other options make it clear that the name has no meaning, here I like v1, v2, ... most as they are short and do not contain (useless) underscores. One could always include the type (or an indicator to the type), e.g. int_1 or i1, i2
81	allow user to rename var name
96	var_ and loc_ might actually be easier to click since it's 5+ chars. "v1", and "a" make the output more concise; using non-numbered variable (a, ...) may run out of symbols for huge functions.
97	In *some* cases with few variables, names such as 'tmp' can be okay, although not nice either. In case the variable is used as a counter/ incrementer/ decrementer, change it to a common counter-name, such as 'i', etc. .
103	nouns with varying lengths
109	Since the order of the variables does not matter the scheme v1, v2, ... does not provide any information beyond "this is a variable". Personally I like to look at the disassembly and the decompilate at the same time. In this scenario something like (v8, vF, ...) could be interesting, referring to the variable offsets ([ebp-8], [ebp-0xF], ...) since it adds information. However this quickly gets impossible when registers are involved.
116	inputs and outputs can be named specially (e.g. scanf input, file input, return values). anything that increments can be named from i, j, k...
120	It really doesn't matter except for cheme 3 :)

Field summary for C7 [1]

Please consider the following parameter naming schemes and rate them from favourite to least favourite: [Ranking 1]

Answer	Count	Percentage
Scheme 1 (A1)	11	29.73%
Scheme 2 (A2)	10	27.03%
Scheme 3 (A3)	2	5.41%
Scheme 4 (A4)	14	37.84%

Field summary for C7 [2]

Please consider the following parameter naming schemes and rate them from favourite to least favourite: [Ranking 2]

Answer	Count	Percentage
Scheme 1 (A1)	10	27.03%
Scheme 2 (A2)	10	27.03%
Scheme 3 (A3)	1	2.70%
Scheme 4 (A4)	16	43.24%

Field summary for C7 [3]

Please consider the following parameter naming schemes and rate them from favourite to least favourite: [Ranking 3]

Answer	Count	Percentage
Scheme 1 (A1)	13	35.14%
Scheme 2 (A2)	15	40.54%
Scheme 3 (A3)	3	8.11%
Scheme 4 (A4)	6	16.22%

Field summary for C7 [4]

Please consider the following parameter naming schemes and rate them from favourite to least favourite: [Ranking 4]

Answer	Count	Percentage
Scheme 1 (A1)	3	8.11%
Scheme 2 (A2)	2	5.41%
Scheme 3 (A3)	31	83.78%
Scheme 4 (A4)	1	2.70%

Field summary for C7F

Do you have any better ideas for parameter naming schemes?

Answer	Count	Percentage
Answer	12	32.43%
No answer	25	67.57%

ID	Response
37	See my answer from the previous questions. Maybe I'd like arg1 more than arg_1, but that's a minor issue. Again, I'm not sure about the animals. Might actually be a good idea (I'm more the visual type of person), but you should evaluate this separately with more samples.
40	param_dog, param_cat, param_wolf, ...
43	Guess the function of the parameters and pick a suitable name :3
49	Again, no, but the parameters should stand out from the local variables.
61	p1, p2, p3... Secretly, I like the emoji scheme most.
81	allow user to rename var name
93	I cannot even read scheme 3, it's rectangles only.
96	same reasons as before
103	nouns with varying length, i.e., dog, cruise, mangosteen, etc
109	No, but the person who came up with the emoji naming should consider calling a therapist. Maybe the police be involved as well.
116	if the type is boolean, flag1, flag2...
120	It does not matter :)

Field summary for C8

Please consider the following two code snippets. Do you prefer reusing variable names for successive for-loops? Reuse No Reuse

Answer	Count	Percentage
I prefer Reuse (A1)	20	54.05%
I prefer No Reuse (A2)	17	45.95%
Comments	13	35.14%
No answer	0	0.00%

ID	Response
37	But really only if the variable is exclusively used as a simple counter in the loop. If you have stunts like in one of the previous samples (the one with the empty body), I'd prefer different names.
49	Other variable names ('jklmn') should be reserved for nested for-loops.
61	Reusing variable names such as i and j is fine. If you do not stick to that naming scheme (why would'nt you?), do not reuse the name. i,j,k can be helpful for nested loops.
69	to have a better clarity on different for loops
75	not a strong preference but reuse helps with scope
81	clearer for analysis
83	Reason: if No Reuse, I have to look carefully to know values of variable i from first loop is not used again. If Reuse, it is obvious.
93	That is not about my preference. Do how it was written in the code. If the author did reuse it, I want reuse. Other, no.
97	Reuse if the first loop-body is not too long.
109	This is probably the only case in which I like variable reuse. Also I would like the loop variables to be scoped in the loop (for(*int* i = 0; ...)). That would make it obvious that its a variable reuse without side-effects.
114	Undecided. Either is fine.
116	Reuse is fine as long as i is never used outside the scope of these loops
120	It does not matter :)

Field summary for D1Q1

What other optimization/de-optimization features would you like to see in your decompiled code? (for example, string/ip address does not appear correctly, thus hard to trace origin, etc.)

Answer	Count	Percentage
Answer	19	51.35%
No answer	18	48.65%

ID	Response
37	Hard to say, I'd have to spend more time decompiling code to actually give an answer here. IP adresses would be great, though! It's hard to convey, but I'd like the output to look like a human could have written the code. In the previous answers this is mostly what I called short and concise. I don't thinks that any human (with a sane mind) would write switches in switches in a do-while loop for instance. So, I'd expect this also not to appear in the output of a decompiler. I'd like to have clean code ;-)
43	sensible names from the start
49	Array-detection String-detection
55	Solving the C++ decompiled code mess, ability to generate vtables and jumping to indirect function calls
58	Nil
61	Minimization of bitwise operations when operating on numbers. Better resolution of function names.
65	dead code elimnation
68	Allow analysts to specify whether to reuse a variable or not (at the variable level, not a global setting). Sometimes reusing variable makes the decompiled cleaner but other times would make it confusing.
77	the decompile string results and the possible values presented in the codes
78	Remove unused variables, dead branches, unused branches
79	Structures.
80	For strings, able to recognize /suggests the language use via the type of character used as string and their appearance frequency Able to import signature to recognized more function
81	to be able to distinguish unicode and ascii through visual emphasis rather than printing out the "raw char/bytes"
93	Resolving API calls correctly is very very very important. Any decompiler not able to resolve ordinal API calls or any API call is useless. Resolve API calls always. Correctly resolving types is also very important. It would be cool if there was struct support, e.g., you can load in custom header files with typedefs, and apply them on your own to variables, and use their fields andsoon.
96	resolution of stack and xmm-based string definitions.
107	IP Address in clear; URL in clear
109	String literals are very important! Not all examples showed them in the decompilate.
116	- Prefer arguments to never change value within a function - Perhaps leaving some decompilation preferences configurable, e.g. threshold for preferring switch-case or if-else, threshold for subexpression length
120	I don't know

Field summary for D1Q2

What quality of life features (for example, GUI interactivity, etc.) do you think would help you use a decompiler much more effectively?

Answer	Count	Percentage
Answer	20	54.05%
No answer	17	45.95%

ID	Response
37	An interactive GUI would be gold! I'd like to be able to rename variables and replace statements with different statements which I find easier to understand. So, basically some simple refactoring features like IDEs provide. Also, other features an IDE provides would be great (code (un-)folding, highlighting a seleced variable etc.). It would also be great to change the representation for single variables, e.g. have one variable as hex and a different one as char (selectable by the user).
40	highlighting (like highlighting all occurences of one variable), additional information on mouse-over (like known variable values?).
43	sensible names from the start
49	Scriptability and thus customizeability integration of my favourite text editor
55	Live group collaboration on the same file, better annotations/note taking, better structure reconstruction
58	for beginner like myself, having GUI or comment as a hint to the data will be beneficial.
65	When some magic numbers accour a hint field could help
68	- Emitting memcpy/memset like what IDA Pro does for snippet of code that perform manual looping (e.g. rep stos on x86) - provide access to decompiler options at function or even more fine-grain level rather than at a global level. - Probably not feasible, but would be nice to allow analyst to modify decompiled code directly so that it remains semantically the same but syntactically different.
77	having a GUI presentation
78	Function tracing
80	Graph representation of the flow of code. Built in "Help" menu and suggestions if user hit a problem
81	allow the user to toggle between the various decompiler parameters that will give rise to the different level of decompiling optimisation to suit the user's feel as different type of codes may be presented better using different params
93	It would be cool if there was struct support, e.g., you can load in custom header files with typedefs, and apply them on your own to variables, and use their fields andsoon.
95	Synchronizing with disassembly view, mouse-over for corresponding register/mem-slot, variable re-naming, copy-paste, concrete and symbolic execution, multiple decompilation candidates.
96	as mentioned earlier, options to control the aggressiveness of optimization/de-optimization might allow an analyst to find a configuration best suited for a given function on a case-by-case basis.
97	Automatically added comments regarding additional parameter-information (as to, in which range they should be) when extracting a method. If in doubt, whether a variable is used as character to int, add the character as comment.
107	GUI - easy with wizard guide
109	In general: Scripting capability. Exposing the AST to a scripting language would be a nice start, but in particular high-level stuff such as listing all local variables with inferred types and where in the AST they are accessed (written or read). And it would be *really* amazing if you could actually execute (parts of) the code from your scripts without having to copy-paste the code into gcc. In particular I'm thinking about string decryption/deobfuscation. Deobfucating strings with inlined decryption routines tends to be painful to automatically decrypt. Admittedly this is not trivial as you would have to allow the script to feed the code with any outside variables (such as globals, parameters and local variables not part of the executed code).
116	Code-editor-like features - Zoom, code-block folding, keyboard shortcuts

	Perhaps colour-coding of arguments (e.g. a1, a2, a3 having different colours in a palette) Allow some aesthetic preferences to be configurable, e.g. variable naming conventions, highlighting/font palette...
120	Jetbrains-style refactorings. Like being able to extract variables or even functionblocks. For example, malware tend to have really long functions. I would love to be able to extract functions from that

Field summary for D1Q3

Is there any particular malware sample (or even a specific function), family, or functionality you would like to see as a decompiled output for the next survey?

Answer	Count	Percentage
Answer	12	32.43%
No answer	25	67.57%

ID	Response
37	I don't want a specific sample here, but I'd like to see more "real-world" code. E.g. network connections and file system interactions and stuff like that.
43	no
58	Nil
65	mirai
68	i've seen assembly code emitted by compiler that tries to align the stack to 16 bytes on every external calls, e.g.: push eax; push eax; push address_of_str1; push address_of_str2; call strcmp Even though strcmp only takes two argument. This can cause decompiler to show redundant arguments or worse, emitting many redundant variables. Would like to see how this can be effectively handled.
77	calling functions within another functions
78	nil
81	nil
93	nope
96	since it's an absolute classic you could go for variants of RC4 implementations. re-using some of the functions/code from the DREAM survey could be interesting to see how things evolved since then.
116	(None I can think of, can I get back to you if I find one?)
120	Mirai's C&C command parsing is fun!