Index	Additional	Description		
0	[2 bytes]	Start of a program line / line number. If an empty line could be a END IF or REPeat or REMark		
***** Operators *****				
1		=	Equal (integer)	See 52, and 53 for ==
2		=	Equal (float)	
3		=	Equal (string)	
4		<>	Not equal (integer)	
5		<>	Not equal (float)	
6		<>	Not equal (string)	
7		<	Less than (integer)	
8		<	Less than (float)	
9		<	Less than (string)	
10		>	Greater than (integer)	
11		>	Greater than (float)	
12		>	Greater than (string)	
13		<=	Less than or equal (integer)	
14		<=	Less than or equal (float)	
15		<=	Less than or equal (string)	
16		>=	Greater than or equal (integer)	
17		>=	Greater than or equal (float)	
18		>=	Greater than or equal (string)	
19		+	Add (integer)	
20		+	Add (float)	
21		-	Subtract (integer)	
22		-	Subtract (float)	
23		*	Multiply (integer)	
24		*	Multiply (float)	
		/	Divide (integer)	Same as 45 - DIV
26		/	Divide (float)	
27		&	Join strings	
28		&&	Bitwise AND	
29		||	Bitwise OR	

30	^^	Bitwise XOR	
40	OR	As in IF (a OR b)	
41	AND	As in IF (a AND b)	
42	XOR		
43	NOT	(Integer)	
44	MOD		
45	DIV	Divide (integer)	
46	NOT	(float)	
47	INSTR		
48	^	Raise to a power (float)	
52	==	Almost equals (float)	May be the same as = (ON)
53	==	Almost equals (string)	

***** Actual values *****

55	[2 bytes]	An actual integer to put on stack	
56	[6 bytes]	An actual (6 byte) floating point to put on stack	
54	[4 bytes]	An actual (4 byte) floating point to put on the stack	
57	[undefined]	An actual string to put on stack	See also 117 for PRINT statements
58		A zero to put on stack (integer)	

***** Normal variables *****

59	[2 bytes]	Get a variable (integer)	
60	[2 bytes]	Get a variable (float)	
61	[2 bytes]	Get a variable (string). Also get an array element	
211	[2 bytes]	Get a sub-string of a string variable (string). Stack contains range	
62	[2 bytes]	Assign a variable (integer)	
63	[2 bytes]	Assign a variable (float)	
64	[2 bytes]	Assign a variable (string)	
210	[2 bytes]	Append a string on the stack onto the end of a string variable	
200	[4 bytes]	Add to an integer variable, an integer	1st word variable, 2nd word value
201	[4 bytes]	Subtract from an integer variable, an integer	1st word variable, 2nd word value
202	[4 bytes]	Multiply an integer variable, by an integer	1st word variable, 2nd word value
205		Add 1 to a float on the stack	
206		Subtract 1 from a float on the stack	

***** Arrays *****

65 [4 bytes] DIMention a integer array (1 or more elements) | First word is no of
66 [4 bytes] DIMention a float array (1 or more elements) | elements - 1
67 [2 bytes] DIMention a string array (1 element)
68 [4 bytes] DIMention a string array (2 or more elements) 1st word is no of elements - 2

69 [2 bytes] Get an array element (integer) multiple element
70 [2 bytes] Get an array element (float) multiple element
Get an array element (string) See 61

71 [2 bytes] Assign a numeric array element (integer)
72 [2 bytes] Assign an array element (float)
73 [2 bytes] Assign an array element (string)
74 [2 bytes] Assign a substring of an array element (string)

***** Stack manipulation *****

75 Covert a string variable on stack to an actual string
76 Convert integer on stack to a float
77 Convert a float to an integer
78 Convert an integer on stack to a string
79 Convert to a negative (integer)
80 Convert to a negative (float)
81 Move a float onto the main stack
82 Move a float from the main stack
83 Convert FP ASCII on stack to a float
84 Convert variable to ASCII for PRINT/INPUT
85 Duplicate integer on top of the stack onto the stack (part of Procedure parameter passing)
86 Move an integer onto the main stack
87 Move an integer from the main stack
88 Convert a decimal ASCII string to an integer (long?)

***** PEEK/POKE *****

90 PEEK
91 PEEK_W
92 PEEK_L
93 POKE
94 POKE_W
95 POKE_L

***** Keyword table commands *****

96		Preceeds actual parameters of a command
97	[2 bytes]	Keyword table entry (procedure)
	[undefined]	Parameter bytes
105	[2 bytes]	Get a variable (integer) for a Procedure call, (Same as 55, but different?)
98	[4 bytes]	Keyword table entry (function)
	[undefined]	Parameter bytes
		1st word - Function type 1=String, 2=Float
		2nd word - Reference to keyword table entry

***** Procedures and Functions *****

100	[4 bytes]	Call a Proc/Fun, also GOSUB	
101	[2 bytes]	Local parameter for proc/fun (integer)	also LOCal variable (integer)
102	[2 bytes]	Local parameter for proc/fun (float)	also LOCal variable (float)
106	[2 bytes]	Local parameter for proc/fun2 (float)	(Same as 102, but different)
103	[2 bytes]	Local parameter for proc/fun ??? string	also LOCal variable (string)
104		Process returned string form a string function call	
108		Something to do with Procedure parameter passing (string)	
109		RETurn/END DEF	

***** PRINT *****

110		PRINT
111		, (comma) In PRINT/INPUT print spaces to the next tab
112		Newline in PRINT/INPUT - On it’s own means PRINT#x
113		TO In PRINT/INPUT
117	[undefined]	An actual string to put on stack for a PRINT
118		! (space separator) In PRINT

***** INPUT *****

120		INPUT (integer)
121		INPUT (float)
122		INPUT (string)
123		, (comma) After text in INPUT/PRINT

***** FOR loops *****

130		Do the FOR	
131	[6 bytes]	Single FOR selection	
132	[4 bytes]	Range FOR selection	
133	[6 bytes]	FOR without STEP	
134	[6 bytes]	FOR with STEP	
		1st word - pointer to loop variable	
		2nd & 3rd words - Pointer to start of FOR loop	
135	[2 bytes]	END FOR	Word is variable

***** IF..THEN *****

140	[2 bytes]	IF/THEN

***** SELECT ON *****

145	[2 bytes]	ON	
146		= (ON) (float)	
147		TO (ON)	
148		= (ON) (integer)	same as index 1?
149		= (ON) (string)	same as index 3?

= REMAINDER is handled by inner loop

***** Various QDOS functions *****

150		CODE()	
151		CHR$()	
152		LEN()	
153		RESPR()	
154		FILL$()	
155		EOF	for embedded DATA statements
156		EOF()	channels
157	[4 bytes]	DIMN	First word is the array Second word is number of elements in the array

***** Various QDOS commands *****

160	[2 bytes]	GOTO watch out for Def Proc/Fun & REPeat & IF/THEN/ELSE
161		STOP also NEW
162		READ integer
163		READ float
164		READ string
165	[4 bytes]	RESTORE
166		CLEAR
167	[8 bytes + (4*number of destinations)]	ON..GOTO
		1st word - Number of destinations
		2nd word - end marker?
		3rd & 4th words - Destination addresses (continues...)
168	[12 bytes + (4*number of destinations)]	ON..GOSUB
		1st word - Number of destinations
		2nd & 3rd words - RETurn addresses
		4th word - end marker?
		5th & 6th words - Destination addresses (continues...)
169	[4 bytes]	EXIT loop, Long GOTO
170	[4 bytes]	NEXT loop, Long GOTO
171		BLOCK

***** Channels *****

180		Check channel is open	(These may be the wrong way round)
181		Check if a channel is a window	

***** Program initialization *****

190		Something to with procedure parameter passing (string)
191		Some kind of a marker

193	[4 bytes]	Used in BASIC program initialization of some sort - Arrays
194	[14 bytes]	Used in BASIC program initialization of some sort
195	[2 bytes]	Used in BASIC program initialization of some sort
196	[2 bytes]	Used in BASIC program initialization of some sort - Floats
197	[2 bytes]	Used in BASIC program initialization of some sort - Integers
198	[2 bytes]	Used in BASIC program initialization of some sort - Floats
199		Used in BASIC program initialization of some sort

***** Turbo Toolkit commands handled internally by the compiler *****

220		POKE$
221		RETRY_HERE
222		MOVE_MEMORY

***** Turbo Toolkit Functions handled internally by the compiler *****

230		PEEK$
231		OPTION_CMD$
232	[2 bytes]	LEN(variable)

***** Odds *****

240	[4 bytes]	Move program pointer to the Long word + A6 (Like GO TO)