

## Programming - a skill for life!

Home | Tutorials | Challenges | Programs | Resources | Glossary

### Appendix to *Let's build a compiler!* containing examples of test programs and test data

1. [Testing Parse01 to Parse12](#)  
Test programs and test data for Parts II and III of *Let's build a compiler!* by Jack Crenshaw
2. [Testing Interpreter01 and Interpreter02](#)  
Test data for Part IV of *Let's build a compiler!* by Jack Crenshaw
3. [Testing Branch01 and Branch02](#)  
Test data and results for Part V of *Let's build a compiler!* by Jack Crenshaw
4. [Testing Bool01, Branch03 and Branch04](#)  
Test program and tests for Part V of *Let's build a compiler!* by Jack Crenshaw
5. [Testing Scanner01 to Scanner04, KISS01 and KISS02](#)  
Test test data and results for Part VII of *Let's build a compiler!* by Jack Crenshaw
6. [Testing TopDown01 and TopDown02](#)  
Examples of test data and results for Part IX of *Let's build a compiler!* by Jack Crenshaw
7. [Testing TINY01 to TINY10](#)  
Test programs and test data for Part X of *Let's build a compiler!* by Jack Crenshaw
8. [Testing TINY11](#)  
Tests for Part XI of *Let's build a compiler!* by Jack Crenshaw
9. [Testing TINY11C, TINY11P and TINY12](#)  
Examples of tests for Part XII of *Let's build a compiler!* by Jack Crenshaw
10. [Testing Procs02 to Procs08](#)  
Examples of tests for Part XIII of *Let's build a compiler!* by Jack Crenshaw
11. [Testing Types01 and Types02](#)  
Examples of tests for Part XIV of *Let's build a compiler!* by Jack Crenshaw
12. [Testing Main02 and Main03](#)  
Test programs and test data for Part XVI of *Let's build a compiler!* by Jack Crenshaw

### Testing TINY11

We copied the generated code into a simple program to test it, then tabulated the results. We provide some examples of the generated code in the table.

Program	Input	Generated Code	Error Message	Result
TINY11	PROGRAM VAR I,PRODUCT BEGIN I=1 PRODUCT=1 WHILE i<5 i=i+1 PRODUCT=PRODUCT*i ENDWHILE END.	Place-holder for MASM start-up code LIB TINYLIB Var I : integer = 0; Var PRODUCT : integer = 0; MAIN: MOV EAX, 1 MOV I, EAX MOV EAX, 1 MOV PRODUCT, EAX @L0: MOV EAX, I PUSH EAX MOV EAX, 5 POP EDX CMP EDX, EAX CMOVL EAX, T CMOVGE EAX, F TEST EAX, -1 JE @L1 MOV EAX, I PUSH EAX MOV EAX, 1 POP EDX ADD EAX, EDX MOV I, EAX MOV EAX, PRODUCT PUSH EAX MOV EAX, I POP EDX IMUL EDX MOV PRODUCT, EAX JMP @L0 @L1: Place-holder for epilog		120
TINY11	PROGRAM VAR INPUT BEGIN READ(INPUT) WRITE(INPUT+1) END.	Place-holder for MASM start-up code LIB TINYLIB Var INPUT : integer = 0; MAIN: CALL READ MOV INPUT, EAX MOV EAX, INPUT PUSH EAX MOV EAX, 1 POP EDX ADD EAX, EDX CALL WRITE Place-holder for epilog		
TINY11	PROGRAM VAR INPUT1 BEGIN READ(INPUT) WRITE(INPUT) END.	Var INPUT1 : integer = 0; MAIN:	Error: Undefined Identifier INPUT.	
TINY11	PROGRAM VAR X,Y,GREATER BEGIN X=(1+2)*3	Place-holder for MASM start-up code I IR TINYI IR		10

	<pre>Y=90/9 IF X&gt;Y   GREATER=X ELSE   GREATER=Y ENDIF END.</pre>	<pre>Var X : integer = 0; Var Y : integer = 0; Var GREATER : integer = 0; MAIN:   MOV EAX, 1   PUSH EAX   MOV EAX, 2   POP EDX   ADD EAX, EDX   PUSH EAX   MOV EAX, 3   POP EDX   IMUL EDX   MOV X, EAX   MOV EAX, 90   PUSH EAX   MOV EAX, 9   MOV ECX, EAX   POP EAX   XOR EDX, EDX   IDIV ECX   MOV Y, EAX   MOV EAX, X   PUSH EAX   MOV EAX, Y   POP EDX   CMP EDX, EAX   CMOVG EAX, T   CMOVLE EAX, F   TEST EAX, -1   JE @L0   MOV EAX, X   MOV GREATER, EAX   JMP @L1   @L0:     MOV EAX, Y     MOV GREATER, EAX   @L1:     Place-holder for     epilog</pre>	
--	---	--	--



[Lexical Scan Revisited](#)

