Decimal	Binary	Hex	Bit #	Decimal
1	**0001**	**1**	1	1
2	**0010**	**2**	2	2
3	**0011**	**3**	3	4
4	**0100**	**4**	4	8
5	**0101**	**5**	5	16
6	**0110**	**6**	6	32
7	**0111**	**7**	7	64
8	**1000**	**8**	8	128
9	**1001**	**9**	9	256
10	**1010**	**A**	10	512
11	**1011**	**B**	11	1024
12	**1100**	**C**	12	2048
13	**1101**	**D**	13	4096
14	**1110**	**E**	14	8192
15	**1111**	**F**	15	16384
			16	32768

1. How many test cases needed to exhaustively test a circuit = **2^(number of inputs)**
2. If binary LC-3 instruction 0010110111111101 or LD R6,DATA is executed at address 0x3050, then what is the address of DATA?
111111101 = -000000011 = -3, so 0x3050 + 1 - 3 = 0x304E
3. ST allows from -256 through +255. **STI, STR** can access full address space. **ADD IM** -16 to +15
4. Opcode extended 5bits = 32 instructions
5.
```
typedef struct
{
char first[32];
char last[32];
int id;
float gpa;
} Student
```
6. `fprintf(FILE* fp,"formatters",values);`
7. Pointers
```
scanf("%s", pstudent->first);
scanf("%f", &(pstudent->gpa));
```
8. Arguments int main(int argc, char *argv[])
argument 1 = argv[1]

9. PUSH	**POP**
ADD R6,R6,#-1	LDR reg,R6,#0
`STR reg,R6,#0`	`ADD R6,R6,#1`

10. Transitors ptype closed without power, open with power. ntype open without power, closed with power
#include <stdio.h> - FILE, stdout, stdin, stderr, putchar, getchar, printf, scanf, fprintf, fscanf,

fopen, fclose, ...
#include <stdlib.h> -atof, atoi, malloc, free, rand, exit, getenv, system, ...
#include <stddef.h> - NULL, size_t, ...
#include <stdbool.h> - bool, true, false
#include <string.h> - memcpy, memset, strcpy, strcat, strlen, strtok, ...
#include <math.h> - sin, cos, tan, exp, log,fmod, fabs, floor, ceil, ...

.1000	.5
.0100	.25
.0010	.125
.0001	.0625

Convert -5.75 to floating Pointers

sign bit = 1 // because its negative
-5 = -101
.75 = .1100
regular binary = -101.1100
move decimal place to make it -1.011100 // 2 shifts
exp = 127 + shifts, exp = 129
129 = 1000 0001 // remove leading 1

Answer: 1 1000 0001 0111 0000 0000 0000 0000 000

Convert 0 10000010 001 1000 0000 0000 0000 0000
Hex: 0x41180000

sign bit = +
exp = 127 + shifts
shifts = exp – 127
shifts = 130 – 127
shifts = 3
// tack 1 back on
1.00110...
shift 3
1001.10....
1001 = 9.5

answer = 9.5

```
FUNCTION ;; stack entry
ADD R6,R6,#-1
ADD R6,R6,#-1
STR R7,R6,#0
ADD R6,R6,#-1
STR R5,R6,#0
ADD R5,R6,#0
LDR R0,R5,#3 ; Body
LDR R1,R5,#4
LDR R2,R5,#5
ADD R3,R2,R1
ADD R3,R3,R0
STR R3,R5,#2
EXIT LDR R5,R6,#0 ; EXIT
ADD R6,R6,#1
LDR R7,R6,#0
ADD R6,R6,#1
RET
```

```
.ORIG x3000
LD R6, STACK
ADD R6,R6,#-1
STR R2,R6,#0
ADD R6,R6,#-1
STR R1,R6,#0
ADD R6,R6,#-1
STR R0,R6,#0
JSR FUNCTION
LDR R3,R6,#0
ADD R6,R6,#4
STACK .FILL x4000
.END
```

Stack Address	Contents
x3FF8	
x3FF9	
x3FFA	Frame Pointer
x3FFB	Return Address
x3FFC	Return Value
x3FFD	Param0
x3FFE	Param1
x3FFF	Param2