Assignment #3

Split the testing set into the following partitions:

Input Condition	**Valid Equiv. Class**	**Invalid Equiv. Class 1**	**Invalid Equiv Class 2**
Size of input string	5 (1)	Not 5 (2)	
1 character in string	True (3)	False (4)	
Character is lowercase	True (5)	False (6)	
'-' present	True (7)	False (8)	
3 numerical chars pres.	True (9)	False(10)	
alphabetic char before '-'	True (11)	False (12)	
'-' char before num chars.	True(13)	False(14)	
Numerical chars sum	10 <= x <= 20 (15)	< 10 (16)	> 20 (17)

Test Case #	**Test Case**	**Covered equivalence classes**
1	"asdE4"	1
2	"asdE"	2
3	"a1234"	3,5,8
4	"ab123"	4
5	"A1234"	6
6	"a-345"	7,9,11,13,15
7	"3-a45"	12,14
8	"a-001"	16
9	"a-999"	17
BVA		
10	"a-111"	Length of input 5
11	"a-11"	Input length 4
12	"a-1111"	Input length 6
13	"a-11"	Numerical char number
14	"a-1111"	Numerical char number

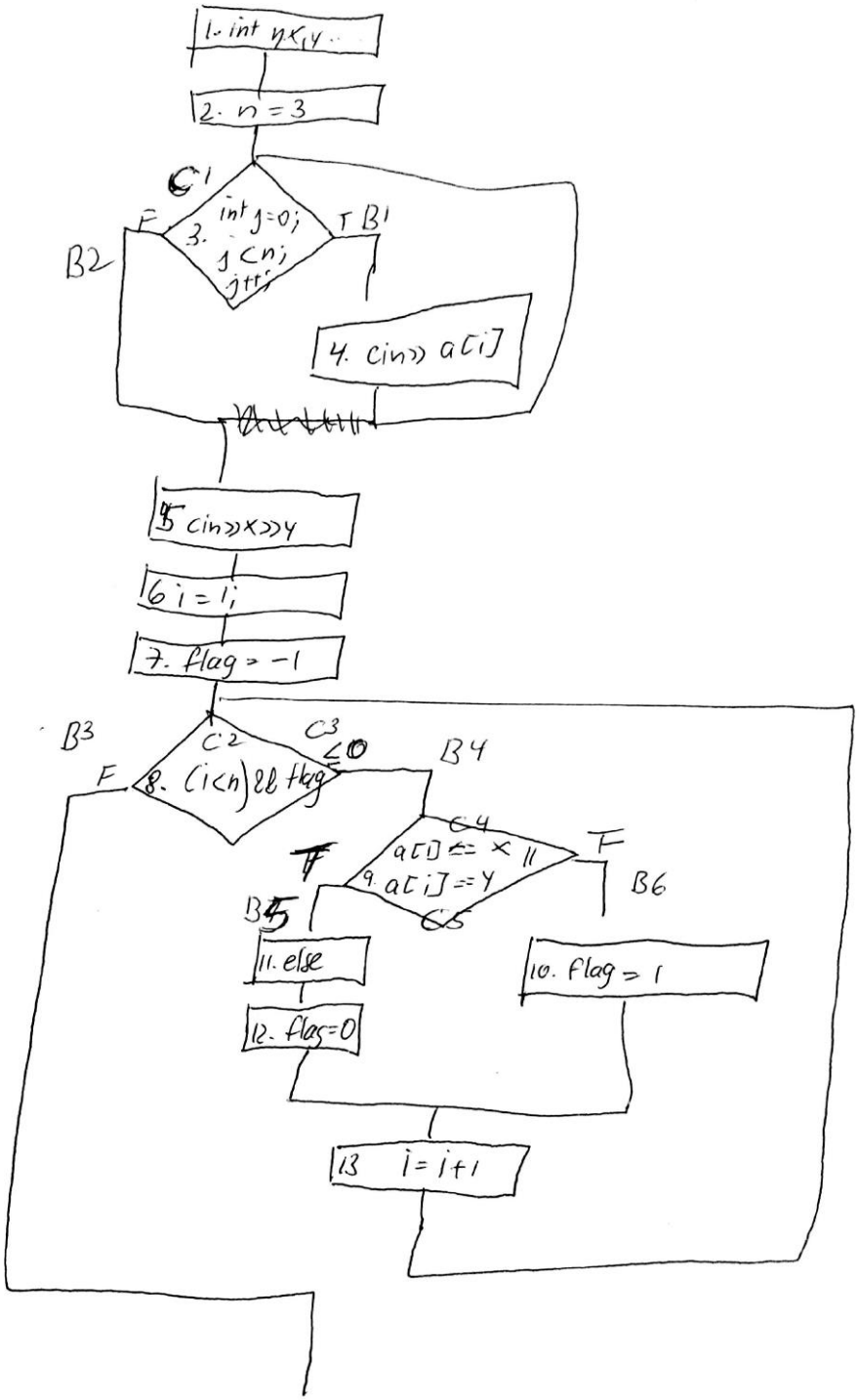
Additional BVA test cases where the sum of all three single digits has to be between 10 and 20 to be a valid number

a-333 => numbers = 9 (invalid)
a-334 -> numbers =10 (valid)
a-335 => numbers = 11 (valid number range)

a-667 => numbers = 19 (valid)
a-668 -> numbers =20 (valid)
a-669 => numbers = 21 (invalid number range)

Assignment #4 #1 Branch testing:

Test case #1: a[0] =1;a[1]=2;a[3]= [3];x=2;y=2; =>Will execute **all** branches: B1, B2, B4, B5, B4, B6, B3



Multiple condition testing:

	C2	C3
a[0] =1, a[1]=2; a[3]= [3],x=2,y=2; (automatically true first iteration)	T	T
a[0] =1, a[1]=2; a[3]= [3],x=2,y=2; (flag set to =1 in first iteration	T	F
a[0] =1, a[1]=2; a[3]= [3],x=4,y=4; (i set to =3 at the end of second iteration	F	T
a[0] =1, a[1]=2; a[3]= [3],x=3,y=3; (i set to =3, flag to 1 at the end of second iteration	F	F

	C4	C5
a[0] =1, a[1]=2; a[3]= [3],x=2,y=2; (First loop iteration)	T	T
a[0] =1, a[1]=2; a[3]= [3],x=2,y=3; (First loop iteration)	T	F
a[0] =1, a[1]=2; a[3]= [3],x=2,y=1; (First loop iteration)	F	T
a[0] =1, a[1]=2; a[3]= [3],x=3,y=3; (First loop iteration)	F	F

#2

Observer pattern: Observers in the face of word counters and file writers and progress displayers would observe the user writing, and be notified whenever a new character is typed.