1) a) Código 1: O erro deste código é que, dentro do “for”, a condição de parada está “i>0” ao invés de “i>=0”, então o índice 0 do vetor não está sendo analisado, no caso mostrado, o erro se dá pelo número procurado se encontrar no ponto 0

Código 2: O erro se dá pela implementação de return, onde, se o valor de x é encontrado em qualquer ponto, ele retorna aquele ponto, ou seja, não está retornando o último valor de X, e sim o primeiro.

Código 3: Primeiramente, há uma ambiguidade na descrição, onde não deixa claro se zero é ou não um número positivo para este problema. No código, podemos ver que o zero está sendo considerado um número positivo, já no teste, não, e este conflito está gerando o erro para esse caso.

Código 4: Além da mesma ambiguidade com o zero do código 3, o erro do código é que a parte do “x[i]%2” deveria ser colocada em valor absoluto, pois se o número for negativo, não resultará em 1, e sim em -1, por isso o caso de teste falhou, pois o “-3” não foi contabilizado

b) Código 1: test: x=[3,5,2]; y=2; Expected =2
Código 2: test: x=[2,3]; Expected = -1
Código 3: test: x=[-4,2,2]; Expected = 2
Código 4: test: x=[1,2,3]; Expected = 3

c) Código 1: Não é possível fazer um teste que exerça o problema e não resulte em falha, pois, se o número procurado estiver na primeira posição(onde se encontra o defeito) o código não o reconhecerá, se não estiver na primeira posição ou não existir, não estará exercitando o defeito
Código 2: test: x=[2, 0, 3]; Expected = 1
Código 3: test: x=[-4, 2, 0, 2]; Expected = 3 (isso se considerarmos que é possível mudar o teste para se encaixar na ambiguidade dada)
Código 4: Não é possível fazer um caso de teste que exercite o defeito mas não dê erro, pois qualquer número negativo, quando aplicado o “%2”, resultará ou em 0 ou em -1, e não será possível testá-lo sem refatorar o código

2) a) A partição “relação entre s1 e s2” satisfaz a propriedade de cobertura/completude, pois cobre todos os possíveis valores que s1 e s2 podem assumir.

b) Não satisfaz a disjunção, pois existem valores de s1 e s2 que se encaixam em mais de um bloco, por exemplo: s1 = {2, 3, 4}; s2 = {2, 3, 4}, estes valores satisfazem três blocos:
representam o mesmo conjunto
s1 é subconjunto de s2
s2 é subconjunto de s1

3) Caminho 1: É possível se bonusPoints =< 0, porém, neste caso, o código sempre retornará um erro (que é o valor setado inicialmente)
Caminho 2: É um caminho impraticável, pois o único caso em que bonusPoints < thresholdJump e bonusPoints > thresholdJump são falsos, é no caso em que

bonusPoints == thresholdJump, e nesse caso, “bonusPoints += 4 * (thresholdJump)” sempre será maior que threshould.
Caminho 5: Entrada: bonusPoints = 50, goldCustomer = true. Percorrerá o caminho mostrado

4) a) Temos, nesse caso, 3 operações diferentes:

Entrada Tique:
- Horario Atual (Entre 00:00 e 23:59)

Saída Tique:
- Horário Atualizado adicionando 1(Entre 00:00:00 e 23:59:59)
- Caso não esteja no intervalo, é feito o rollover para o próximo dia, ou seja, reseta a contagem

Entrada Compare:
- Dois horários a serem comparados

Saída Compare:
- Resposta referente ao resultado, ou seja, qual é o anterior ou se é igual
- Caso um horário ultrapasse para o início do outro dia, será ainda considerado menor(Por exemplo, passar para 00:00:00, será considerado menor que 23:59:59)

Entrada Add:
- Dois horários a serem calculados

Saída Add:
- Soma entre os horários

Entrada Subtraction:
- Dois horários a serem calculados

Saída Subtraction:
- Diferença entre os horários
- Sempre o relógio atual será subtraído pelo outro

b)
Casos de teste para Tique:
Horários Válidos:
- Entrada: 00:00:00
- Saída: 00:00:01

Rollover segundos:
- Entrada: 00:00:59
- Saída: 00:01:00

Rollover minutos:
- Entrada: 00:59:59
- Saída: 01:00:00

Reset de horas:
- Entrada: 23:59:59
- Saída: 00:00:00

Horários inválidos:
- Entrada: 24:30:00
- Saída: 00:30:01

Casos de teste para Compare:

A=0 && B=0:
- Entrada: A= 00:00:00 / B = 00:00:00
- Saída: 0

B>A:
- Entrada: A = 12:30:20 / B = 13:50:33
- Saída: -1

A>B:
- Entrada: A = 16:30:26 / B = 13:50:49
- Saída: 1

A=B:
- Entrada: A = 12:30:00 / B = 12:30:00
- Saída: 0

Casos de teste para Add:

Ambos 0:
Entrada: A = 00:00:00 / B = 00:00:00
Saída: 00:00:00

Caso sem Rollover:
Entrada: A = 12:30:20 / B = 00:20:00
Saída: 12:50:20

Rollover Segundos:
Entrada: A = 12:30:00 / B = 00:30:00
Saída: 13:00:00

Rollover Minutos:
Entrada: A = 12:30:00 / B = 00:30:00
Saída: 13:00:00

Rollover Horas:
Entrada: A = 12:30:00 / B = 12:00:00
Saída: A = 00:30:00

Casos de teste para Subtraction:

Ambos 0:
Entrada: A = 00:00:00 / B = 00:00:00
Saída: 00:00:00

A>B:
Entrada: A = 12:30:00 / B = 12:20:00
Saída: 00:10:00

B>A:
Entrada: A = 00:40:00 / B = 01:00:00
Saída: 23:40:00

B=A:
Entrada: A = 12:30:00 / B = 12:30:00
Saída: 00:00:00