5) For the following questions, if your answer is YES, you need to provide justification. If your answer is NO, you need to give a counterexample proving why the statement is not true. An answer without any explanation receives zero point.

a) Does 100% ***decision*** coverage always give 100% ***block*** coverage? (5 points)
YES
The blocks not in branches will be covered by any test, all other blocks are in certain branch.

b) Does 100% ***block*** coverage always give 100% ***decision*** coverage? (5 points)
NO

```
1    begin
2      int x;
3      input(x);
4      if(x < 0)
5        x = -x;
6      output(x);
7    end
```

t: <x = -3> can achieve 100% block coverage but only 50% decision coverage, the FALSE branch of decision in line 4 is not covered.

c) Does 100% ***simple condition*** coverage always give 100% ***block*** coverage? (5 points)
NO

```
1    begin
2      int x, y, z;
3      input(x, y);
4      if(x < 0 or y < 0)
5        z = foo(x, y);
6      else
7        z = bar(x, y);
8      output(z);
9    end
```

t1: <x = -3, y = 2> and t2: <x = 3, y = -2> can achieve 100% simple condition coverage but not 100% block coverage, block {line 6, line 7} is not covered.

d) Does 100% ***block*** coverage always give 100% ***simple condition*** coverage? (5 points)
NO

```
1    begin
2      int x, y, z;
```

```
3       input(x, y);
4       if(x < 0 or y < 0)
5          z = foo(x, y);
6       else
7          z = bar(x, y);
8       output(z);
9    end
```

t1: <x = -3, y = 2> and t2: <x = 3, y = 2> can achieve 100% block coverage but not 100% simple condition coverage, TRUE value of condition y < 0 is not covered.

e) Does 100% ***simple condition*** coverage always give 100% ***decision*** coverage? (5 points)
NO

```
1    begin
2       int x, y, z;
3       input(x, y);
4       if(x < 0 or y < 0)
5          z = foo(x, y);
6       else
7          z = bar(x, y);
8       output(z);
9    end
```

t1: <x = -3, y = 2> and t2: <x = 3, y = -2> can achieve 100% simple condition coverage but only 100% decision coverage, the FALSE branch of decision in line 4 is not covered.

f) Does 100% ***decision*** coverage always give 100% ***simple condition*** coverage? (5 points)
NO

```
1    begin
2       int x, y, z;
3       input(x, y);
4       if(x < 0 or y < 0)
5          z = foo(x, y);
6       else
7          z = bar(x, y);
8       output(z);
9    end
```

t1: <x = -3, y = 2> and t2: <x = 3, y = 2> can achieve 100% decision coverage but not 100% simple condition coverage, TRUE value of condition y < 0 is not covered.

6) Consider the following program that is supposed to check if the input data item is in the range 0 to 100, inclusive

```
int check(x);
int x;
{
if ((x>=0 )&& (x<=200))
check=true;
else check=false;
}
```

a) Find a test set T which is adequate with respect to the **block coverage** and **does not** reveal the error (5 points)

T = {t1: <x = 50>; t2: <x = 300>}

b) Find a test set T which is adequate with respect to the **block coverage** and **reveals** the error (5 points)

T = {t1: <x = 150>; t2: <x = 300>}

c) Find a test set T which is adequate with respect to the **simple condition coverage** and **does not** reveal the error (5 points)

T = {t1: <x = 50>; t2: <x = 300>; t3: <x = -1>}

d) Find a test set T which is adequate with respect to the **simple condition coverage** and **reveals** the error (5 points)

T = {t1: <x = 150>; t2: <x = 300>; t3: <x = -1>}

e) Find a test set T which is adequate with respect to the **decision coverage** and **does not** reveal the error (5 points)

T = {t1: <x = 50>; t2: <x = 300>}

f) Find a test set T which is adequate with respect to the **decision coverage** and **reveals** the error (5 points)

T = {t1: <x = 150>; t2: <x = 300>}

g) Find a test set T which is adequate with respect to the **multiple condition coverage** and **does not** reveal the error (5 points)

T = {t1: <x = 50>; t2: <x = 300>; t3: <x = -1>}

h) Find a test set T which is adequate with respect to the **multiple condition coverage** and **reveals** the error (5 points)

T = {t1: <x = 150>; t2: <x = 300>; t3: <x = -1>}