Pb 1:

(a)

To prove the logical equivalence relation, ≡, is an equivalence relation on F, we should prove: ≡, contain the property of Reflexivity, Symmetry, Transitivity: on F

Reflexivity:

We suppose there are one formula $\phi \in$ F, obviously $\phi = \phi$, v(ϕ) = v(ϕ), therefore $\phi \equiv \phi$ so (ϕ,ϕ) $\in$ R, it is reflexivity.

Symmetry:

we suppose there are two formulas ϕ and $\varphi \in$ F, if (ϕ,φ) $\in$ R, then v(ϕ) = v(φ), therefore $\phi \equiv \varphi$. Because v(φ)= v(ϕ), therefore (φ,ϕ) $\in$ R, so it satisfy symmetry.

Transitivity:

We suppose there are three formulas x, y, z $\in$ F, if (x, y)$\in$ R, v(x) = v(y), x ≡ y,
If (y, z)$\in$ R, v(y) = v(z) , y ≡ z, because v(x) = v(z), x ≡ z, therefore (x, z)$\in$ R, so it satisfy transitivity.

(b)

Because F denote the well-formed formula, suppose p, q are propositional variables in F

[⊥] = { ⊥, p ∧¬ p, p∧⊥, q ∧¬ q, q∧⊥, ⊥∨⊥}

(c)

(i) for all ϕ, ϕ'and φ, φ'$\in$ F , because $\phi \equiv \phi$', therefore v(ϕ) = v(ϕ')

Because ! v(ϕ) =! v(ϕ') , therefore v(¬ϕ) = v(¬ϕ'), so ¬ϕ ≡ ¬ϕ'.

(ii) Because v($\phi \wedge \varphi$) = v(ϕ) && v(φ)

v(ϕ'$\wedge \varphi$') = v(ϕ') && v(φ')

because $\phi \equiv \phi$'; $\varphi \equiv \varphi$', therefore v(ϕ) = v(ϕ'); v(φ) = v(φ')

so v($\phi \wedge \varphi$) = v(ϕ'$\wedge \varphi$'), therefore $\phi \wedge \varphi \equiv \phi$'$\wedge \varphi$'

(iii) Because v($\phi \vee \varphi$) = v(ϕ) || v(φ)

$$v(\phi' \vee \varphi') = v(\phi') \,\|\, v(\varphi')$$

$$v(\phi) = v(\phi'); v(\varphi) = v(\varphi')$$

so $v(\phi \vee \varphi) = v(\phi' \vee \varphi')$, therefore $\phi \vee \varphi \equiv \phi' \vee \varphi'$

(d)

We know that a Boolean structure contain {T, $\vee$, $\wedge$, ', T, $\perp$, **zero, one}**

$F_{\equiv}$ together with the operations defined above forms a Boolean Algebra, the new Boolean algebra must satisfy the following laws: commutative, associative, distributive, identity, complementation.

we can use the law of complementation: to get the zero and one in $F_{\equiv}$:

Because $x \vee x' = 1$,

$[\phi] \vee [\neg\phi] = 1$

$[\phi \vee \neg\phi] = [\,T\,]$

So one is [T] , which means one is a set of equivalence class, which is always true.

Because $x \wedge x' = 0$

$[\phi] \wedge [\neg\phi] = 1$

$[\phi \wedge \neg\phi] = [\perp]$

So the zero is $[\perp]$, which means one is a set of equivalence class, which is always false.

The proof for $F_{\equiv}$ formed a Boolean Algebra as follow:

Commutative:

Suppose [x], [y] are two elements in F:

$[x] \vee [y] = [x \vee y] = [y \vee x] = [y] \vee [x]$

Therefore $[x] \vee [y] = [y] \vee [x]$

The similar for the $[x] \wedge [y] = [y] \wedge [x]$

Associative:

Suppose [x], [y], [z] are three elements in F:

$([x] \vee [y]) \vee [z]$

$= ([x] \vee [y] \vee [z])$

$= [x] \vee ([y] \vee [z])$

Therefore $([x] \vee [y]) \vee [z] = [x] \vee ([y] \vee [z])$

The similar for the $([x]\wedge[y])\wedge[z] = [x]\wedge([y]\wedge[z])$

Distributive:

Suppose [x], [y], [z] are three elements in F:

$[x]\vee([y]\wedge[z]) = [x]\vee([y\wedge z]) = [x\vee(y\wedge z)]$
$= [(x\vee y)\wedge(x\vee z)] = [x\vee y]\wedge[x\vee z]$
$= ([x]\vee[y])\wedge([x]\vee[z])$
Therefore $[x]\vee([y]\wedge[z]) = ([x]\vee[y])\wedge([x]\vee[z])$
The similar for the $[x]\wedge([y]\vee[z]) = ([x]\wedge[y])\vee([x]\wedge[z])$

Identity:

Suppose [x], is elements in F:
$[x]\vee[\perp] = [x\vee\perp] = [x]$
$[x]\wedge[\,T\,] = [x\wedge T] = [x]$

Complementation:

Suppose [x], is elements in F:

$[x]\vee[x'] = [x\vee x'] = [T]$

$[x]\wedge[x'] = [x\wedge x'] = [\perp]$

In conclusion, $F_{\equiv}$ together with the operations defined above forms a Boolean Algebra.

Pb 2:

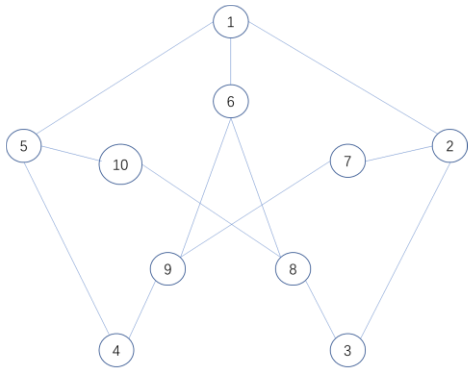
(a)

We know that for K_5, every vertex must has four edges. And to get the subdivision of Petersen, we can use strategy by reducing vertex or edges ,but in this progress, the degrees of each vertex will only get smaller, and for Petersen the number of each vertex is at most 3, which means whatever we change the Petersen, we cannot get subdivision which contain K_5.

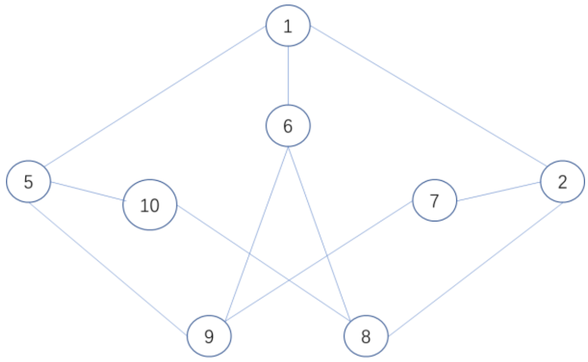
(b)

To show that the Petersen graph contain a subdivision of $K_{3,3}$, we can use strategy by reducing vertex and edges to get a $K_{3,3}$ graph from original Petersen graph; the process is as follows:

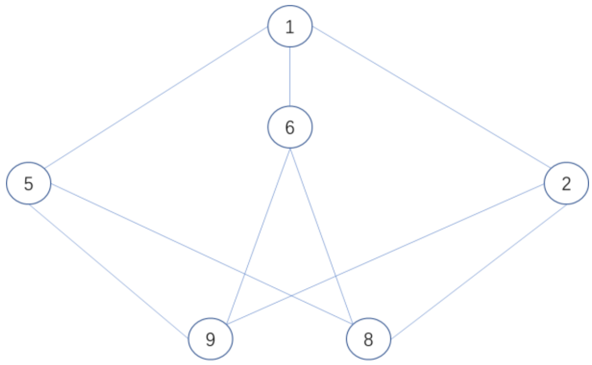
(1) Firstly, remove edge 7-10 and edge 3-4



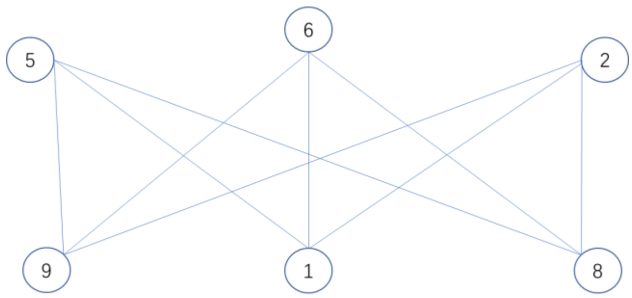
(2) then remove vertex (4) and vertex(3), connect vertex(5) to vertex(9) with edge; and connect vertex(2) to vertex(8) with edge.



(3) finally, remove the vertex (10) and vertex(7), connect vertex(5) to vertex(8) with edge; and connect vertex(2) to vertex(9) with edge.



Then the graph above is $K_{3,3}$, which is isomerize with the graph under. In conclusion, Petersen graph contains a subdivision of $K_{3,3}$



Pb3:

(a)

In this problem we can define different classes as different vertices:

Defence against the DarkArts as Vertex(D),

Potions as Vertex(P),

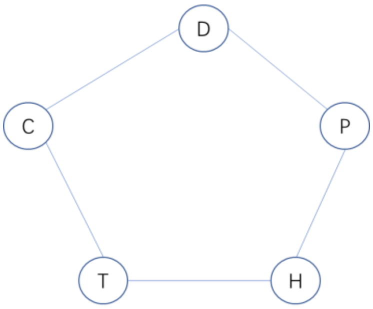
Herbology as vertex(H),

Transfiguration as vertex(T),

Charms as vertex(C);

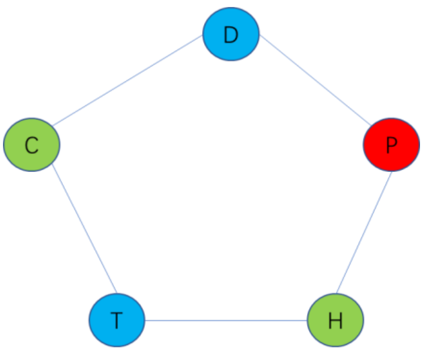
And the edge between two vertices means two subjects are conflict.

According to the graph we defined, the problem of finding the maximum number of classes can transfer into coloring the graph; which means the vertex with same color are the classes Harry can take at the same time. Therefore, by counting the number of the same color, we can find the best choice for Harry. The definition above can draw the graph as follow:



(b)

Suppose we color *Charms* and *Herbology* with green, *Defence against the DarkArts* and *Transfiguration* with blue, *Potions* with red, we can get the following graph:



In conclusion, at this situation, the maximum number of classes Harry can take is two, the choice is either *Charms* and *Herbology* or *Defence against the DarkArts* and *Transfiguration*.

However, at different strategy of coloring, we may also get choices such as *Charms and Potions* *or Transfiguration and Potions or Defence against the DarkArts* and *Herbology;* and the maximum number is still two.

Pb 4:

(a)

According to the condition given, T(n) denote the number of binary trees with n nodes. We set T(x,y) to show the in a binary tree, the left tree contain x nodes, and the right tree contain y nodes.

Therefore T(n) = T(0, n-1)+T(1, n-2)+T(1, n-3)+……+T(n-1, 0)

And we can simplify the T(x, y), because the number of T(x, y),is same as the number of T(x) multiply number of T(y), just like: T(x, y)= T(x) ×T(y)

Therefore T(n) = T(0)×T(n-1)+T(1)×T(n-2)+T(2)×T(n-3)+……+T(n-1)×T(0)

$$T_{(n)} = \sum_{i=0}^{n-1} T_i T_{(n-1-i)}$$

(b)

According to the definition given, the smallest binary tree is a node with two children, we can treat it as a base case, and the number of node in base case is always odd, and if we want to build a full binary tree, we should add 2n (n∈Z) leaves on one node, because the number of every recursion is even, therefore the sum of node is always odd, because of the base case. Therefore, a full binary tree must have an odd number of nodes.

(c)

From the T(n) calculated from question (b), we can get the number of binary tree with different n, and we can easily calculate the B(n), when n is smaller than 7, so we can get the follow excel:

	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7
T(n)	1	1	2	5	14	42	132	429
B(n)	0	1	0	1	0	2	0	5

We can find that T(n’) = B(2n+1), when n’≤n. we can explain in that way: In B(n), because it is a full binary tree, it can only add two nodes at the same time under one node, which is similar like T(n’) add one node under one node. Furthermore, when n = 0, there is no full binary tree according to definition, therefore T(n’) = B(2n’+1)

(d)

If it has n propositional variables, from the conclusion in last assignment, the number of internal node is n-1, if the node is n. Because the F(n) is a well-formed formula in Negated normal form, therefore we can treat $\wedge, \vee$ as node to connect the propositional variables(treat as leaves), therefore the number of the possible of node is 2^{n-1} and possible leaves is n!.

Therefore $F(n) = B(2n-1) \times 2^{n-1} \times n!$

Pb 5:

(a)

To take p1(n+1) as an example: p(n+1) can only get from v2 and v4 by one step(from p(n)), therefore the probability of v4 move to v1 is 1/3 which is the same for v4 move to v1, therefore the p1(n+1) is the probability of p2(n) multiply by 1/3, the similar for p4(n). so we can get the following formula:

$$p1(n+1) = \frac{1}{3}p2(n) + \frac{1}{3}p4(n)$$

$$p2(n+1) = \frac{1}{2}p1(n) + \frac{1}{2}p3(n) + \frac{1}{3}p4(n)$$

$$p3(n+1) = \frac{1}{3}p2(n) + \frac{1}{3}p4(n)$$

$$p4(n+1) = \frac{1}{2}p1(n) + \frac{1}{2}p3(n) + \frac{1}{3}p2(n)$$

(b)

Because the steady state can be determined by *pi(n +1) = pi(n),* we can get the following formulas:

$$p1(n) = \frac{1}{3}p2(n) + \frac{1}{3}p4(n)$$

$$p2(n) = \frac{1}{2}p1(n) + \frac{1}{2}p3(n) + \frac{1}{3}p4(n)$$

$$p3(n) = \frac{1}{3}p2(n) + \frac{1}{3}p4(n)$$

$$p4(n) = \frac{1}{2}p1(n) + \frac{1}{2}p3(n) + \frac{1}{3}p2(n)$$

And we know that the sum of each event must be 1, so:

$$p1(n) + p2(n) + p3(n) + p4(n) = 1$$

After solving the five equations above, we can get:

p1(n) = 1/5 ; p2(n) = 3/10

p3(n) = 1/5 ; p4(n) = 3/10

(c)

We use v1v2 show that the shortest distance between v1 and v2, the same to v1v2, v1v3, v1v4; and we suppose the distance between the adjacent vertex is **1**, so we can get the following formula:

$$E = \frac{1}{5} v1v1 + \frac{3}{10} v1v2 + \frac{1}{5} v1v3 + \frac{3}{10} v1v4$$

$$= \frac{1}{5} * 0 + \frac{3}{10} * 1 + \frac{1}{5} * 2 + \frac{3}{10} * 1$$

$$= 1$$

Therefore, the expected distance from v1 in the steady state is 1.