GITHUB UPLOADED ANSWERS FOR EXAM – THIS WAS UPLOADED 2 DAYS AFTER THE DEADLINE @4.01pm 09/05/2020

Discrete Maths: Daniel Nugent

Question 1

a

i)

Show that $a \sim b$, we must show that $\sim$ is reflexive, symmetric and transitive.

$a = b + pq$ for some $q \in P$.

Reflexivity: $a \sim a$

$q = 0$ as $0 \in P$. (0 polynomial)

$a = a + p(0)$

$a = a$

Symmetric: $a \sim b \Rightarrow b \sim a$

$a = b + pq' \Rightarrow b = a + pq$

$b = a - pq'$

$a - pq' = a + pq$

$-pq' = pq$

q' must be $= -q$ which is an element of P.

Reflexivity: $a \sim b \wedge b \sim c \Rightarrow a \sim c$

$b = a - pq'$

$b = c + pq$

$a - pq' = c + pq$

$a = c + p(q + q')$

Summation of two polynomials in P, is an element of P, transitivity holds.

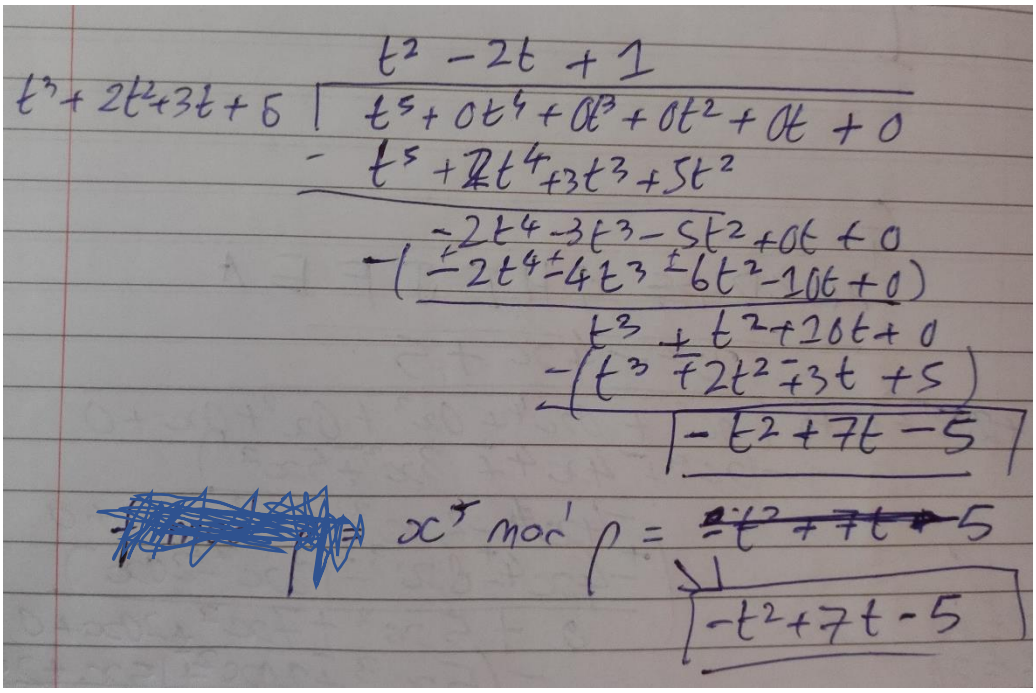
ii)

As the equivalence relation $a \sim b$ is defined as if $a = b + pq$, we can assume that unless $b = 0$ (which only occurs in one case, and in that case the unique representative would be 0) p doesn't divide in equally to a. Hence there will be a remainder, which can be found from calculating a mod p. From the algebraic long division, the remainder will never have a t component with an exponent > 2 as the highest exponent in p is 3. Therefore, each equivalence class must have a unique representative of degree less than 3.

iii)

To find the unique representative we look at the equivalence relation and it's fairly straightforward. From a = b + pq we see that we are looking for the components of b with an exponent less than t^3. By using long division, we can find q and using the remainder (modular arithmetic, we can find b).

So for any equivalence class of say a = {x ∈ P | x ~ a}, in this case a being t^5 we must find t^5 *mod* $(t^3 + 2t^2 + 3t + 5)$. I did the long division on paper below



I got the unique representative in t^5's equivalence class to be $-t^2 + 7t - 5$.

b)

(a)

If set S has m elements and set T has n elements and m < n, then there are clearly more elements in T. φ maps S -> T. The function φ therefore cannot be a bijection as there will be elements in the codomain with no mapping from S. And as φ is a function, each element in the domain (S) can only map to at most one element. Hence there will be some element w ∈ T, such that no u ∈ S, φ(u) = w. *QED.*

(b)

In this case m > n, so there are more elements in S. As every element in S maps to T, and there are more elements in S, there must be some elements in the domain which map to the same element in the codomain. Let those elements be u, v ∈ S, u != v, φ(u) = φ(v). *QED*

<u>Question 2</u>

a

i)

Need to show that $*$ is an associative binary operation on A.

$\forall a, b, c \in A,\ a * (b * c) = (a * b) * c$

$\forall a, b \in A, (a * b) \in A$. Hence it is a binary operation (From the table no result is greater than 3 and all results are elements of A itself).

From the table in the paper, $*$ is basically multiplication with that condition that the result cannot be greater than 3. So, it subtracts 4 indefinitely until the result is less than or equal to 3. For example, $3 * 3 = 9$. So, we subtract 4. $9 - 4 = 5$. Again. $5 - 4 = 1$. Now that is the result. As we already know multiplication is associative. We can clearly see that $*$ is also associative. Hence $(A, *)$ is a semigroup.

Now let's see if it's a monoid. For it to be a monoid it needs to have an identity element e such that

$\forall a \in A, a * e = a = e * a$

As with multiplication, 1 is also the identity element e here. As there are no elements in A greater than 3, we will have no issue with the result changing. Hence $(A, *)$ is a monoid.

Let's check if it is a group. All elements in A must be invertible for $(A, *)$ to be a group. $\forall a, b \in A, a * b = b * a = e$. b is said to be the inverse of a here. a is said to be invertible. Assume it is a group. $a * b = b * a = 1$ must hold true.

The only two results from our table that result in 1 are $1 * 1$ and $3 * 3$. Therefore, it is not group as if $a = 0$ or $a = 2$, there is no inverse as there is no b such that $0 * b = 1$ or $2 * b = 1$.

That means if a were 0 or 2 there would be no inverse of it. Therefore $(A, *)$ cannot be a group.

b) It is a semigroup as the associative property still applies, but also the binary operation property.

$\forall a, b \in A, (a * b) \in A'$, where $A' = (\{1,3\}, *)$

$1 * 1 = 1 \in A'$

$1 * 3 = 3 \in A'$

$3 * 1 = 3 \in A'$

$3 * 3 = 1 \in A'$

It is also a monoid as it has an identity element e, $= 1$, which we can clearly see above and still holds true.

It is however also a group.

The only two equations that result in 1 (the identity element) are once again

$1 * 1 = 1$ and $3 * 3 = 1$. Therefore, it is a group as 0 and 2 are omitted in this subset. 1 is the inverse of 1, and 3 is the inverse of 3.

c) It is a semigroup as the associative property still applies, but also the binary operation property.

$\forall a, b \in A''$, $(a * b) \in A''$, where $A'' = (\{0,1,3\}, *)$. All the results of $a * b$ are still elements of $\{0,1,3\}$.

It is also a monoid as it has an identity element $e = 1$, which still holds true. It is not however a group. For it to be a group $a * b = b * a = 1$ must hold true. The only two equations that result in 1 (the identity element) are once again

$1 * 1 = 1$ and $3 * 3 = 1$. Therefore, it is not group as if $a = 0$, there is no inverse as there is no b such that $0 * b = 1$. That means if a were 0 there would be no inverse of it. Therefore $(A'', *)$ cannot be a group.

d)

- x -> x
- x -> 1
- x -> 0
- x -> x * x

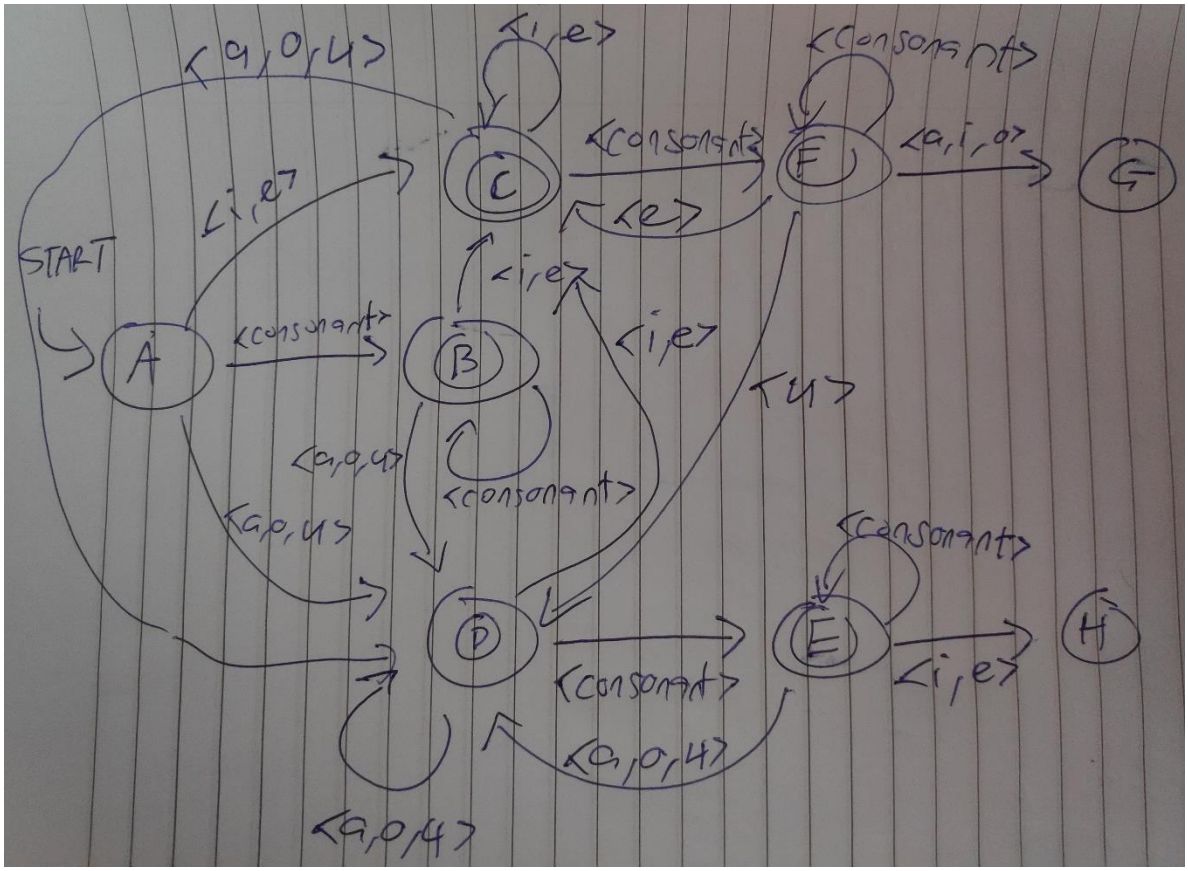
e)

We already know from lectures than composition of multiple functions is associative.

$(C, \circ)$ is a semigroup as $\circ$ is a binary operation and associative. Mapping the input to the output directly ($\varphi(x)$ -> x), is the identity element e in the case of $(C, \circ)$. A semigroup with an identity element e such that $a \circ e = a = e \circ a$, is a monoid. It is given in the question that C is the set of invertible homomorphisms from B to itself, hence C is invertible. Therefore $(C, \circ)$ is a group.

Question 3

a)



b)

The language L is infinite. In states C, E, F we can recursively concatenate a consonant to the word indefinitely. Hence $|L| = \infty$.

The language L must be regular as we can draw a finite state automaton which recognises it.

The language L is context-free.

~~We can use the pumping lemma in an attempt to pump the language to prove this.~~

~~If L is a context-free language, L has a pumping length P such that any string S where $|S| \geq P$ may be divided into 5 pieces. S = uvxyz such that the following are true~~

1. ~~$uv^i xy^i z$ is in L for every $i \geq 0$.~~
2. ~~$|uv| > 0$.~~
3. ~~$|vxy| \leq P$.~~

~~Assume we have some string S ∈ L = 'iaibbee'. Assume pumping length is 7. S is clearly in L and satisfies the rules of the language.~~

~~Let u = 'i'. Let v = 'aib'. Let x = 'b'. Let y = 'e' and let z = 'e'. Currently we have i = 1, but let's make i = 2 and see if our string S ∈ L. S' = ''. S' ∉ L as we have ''iaibaibbeee' and we breach the rule that 'i' followed by a string of consonants followed by an 'a' at 'iba'.~~

~~However, let's try another way of destructuring the word.~~

~~Let u = 'ia'. Let v = 'ib'. Let x = 'b'. Let y = 'e' and let z = 'e'. Currently we have i = 1, but let's make i = 2 and see if our string S ∈ L. S' = ''. S' ∈ L as we have ''iaibibbeee' and we don't breach any of the rules of the language.~~

The language must be context free as it is regular as it can be accepted by an FSA.

c)

minimal: {a, b, c, d, e, f, g, h, i, l, m, n, o, p, r, s, t, u}

least: none

d)

As we have already said that the language L is regular, we need to show that there are finite many equivalence classes using the Myhill-Nerode theorem.

$x \sim y \leftrightarrow \forall z \in A^* (xz \in L \leftrightarrow yz \in L)$.

We can find the equivalence classes by examining the states in the FSA which we have already created.

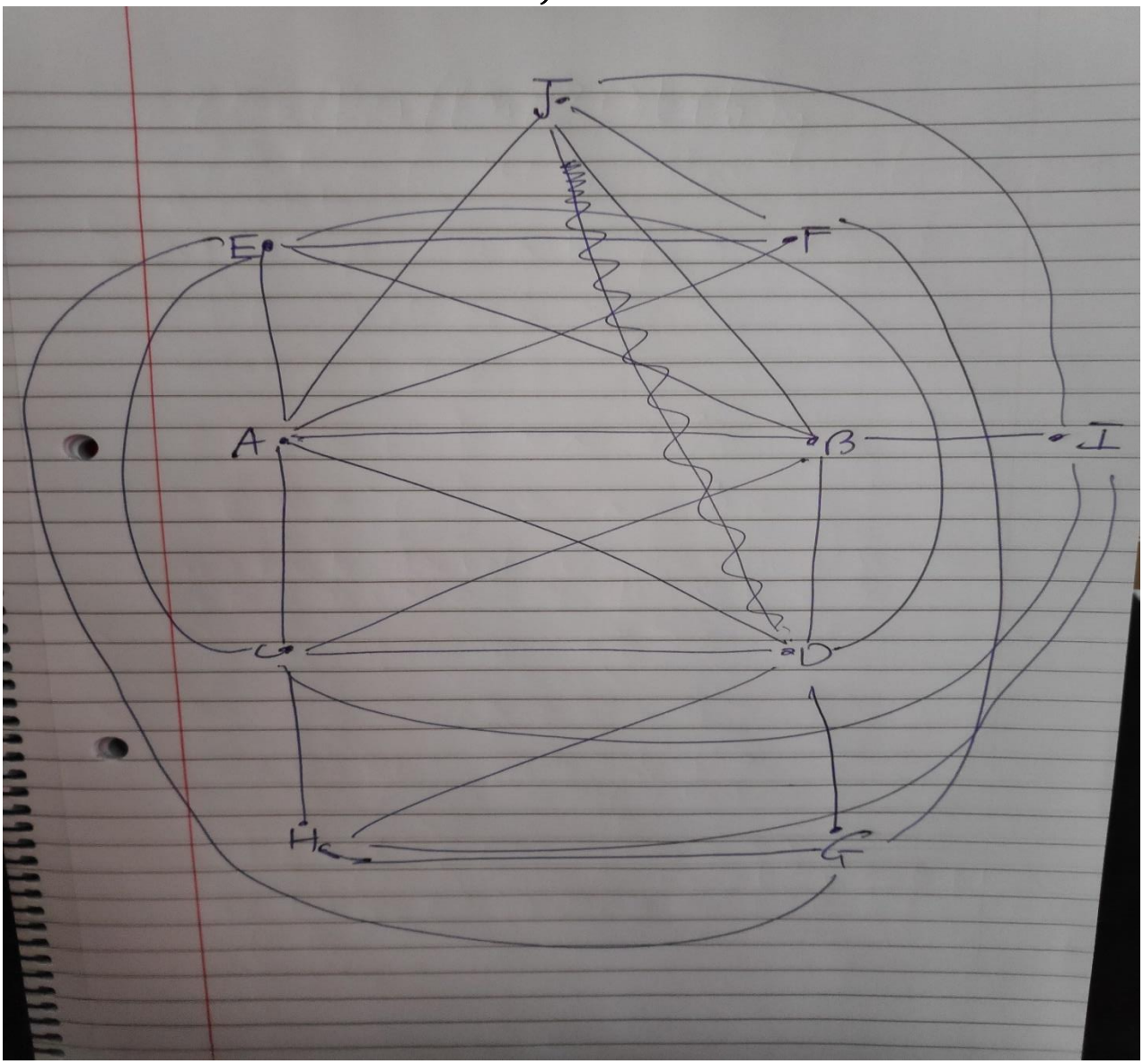
1. (State E is) the state where the last letter concatenated to the word is <i, e>.
2. (State C is) the state where the last letter concatenated to the word is <a, o, u>.
3. (State G is) the state where the <i, e> was concatenated and then a string of one or more consonants was concatenated (last letter is a consonant, but last vowel was <i,e>).
4. (State F is) the state where the <a, o, u> was concatenated and then a string of one or more consonants was concatenated (last letter is a consonant, but last vowel was <a,o,u>).
5. (State D is) the state where no vowels have been concatenated to the word yet (only consonants).

I found in total 5 equivalence classes, which shows that L is regular as finite many equivalence classes (we already knew that).

Question 4

a)

i)



ii)

20 edges. We need 45 edges in total to make a graph with 10 edges complete and we currently have 25, hence we need 20 (1+2+3+….+9).

iii)

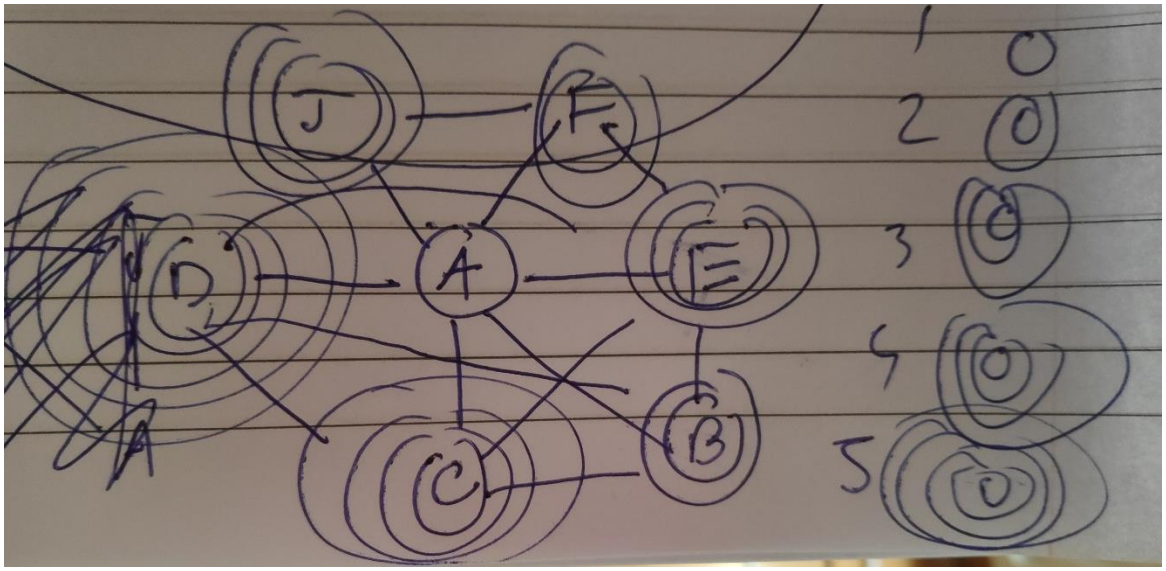
In a regular graph, each vertex has the same degree. To make this graph regular we remove edges until each vertex has a degree of 4. We must remove the edges CD, BC, AB, AE, DE.

iv)

As the lowest degree of any vertex is 4, we must remove at least 4 vertices to make it disconnected.

v)

5 distinct colours. Take A for an example as it has the highest degree. A is connected to B, C, D, E, J, F. But those vertices are also interconnected. I drew a diagram to make it easier to see. The number of circles is different colours. D has 5 circles.



vi)

Yes, as $\forall v \in V$, deg(v) is even. E.g.: deg(a) = 6, deg(b) = 6, deg(c) = 4 and so on.

vii)

We know there is a Hamiltonian circuit if the graph is bipartite and is in the form $K_{p,q}$, where $|p - q| \leq 1$. Unfortunately, Hamiltonian circuits are an unsolved problem so even if this can't be achieved, there may still be Hamiltonian circuits. One such example I was able to find was: ABCDGHIJFEA

b)

There are 10 isomorphisms in total. We can rearrange the diagram as a pentagon (and inside of it another pentagon), so thus being 5 lines of symmetry and we multiply that number by 2 as we can flip each line of symmetry either way.

1
A=E
B=D
C=C
D=B
E=A
F=F
G=J

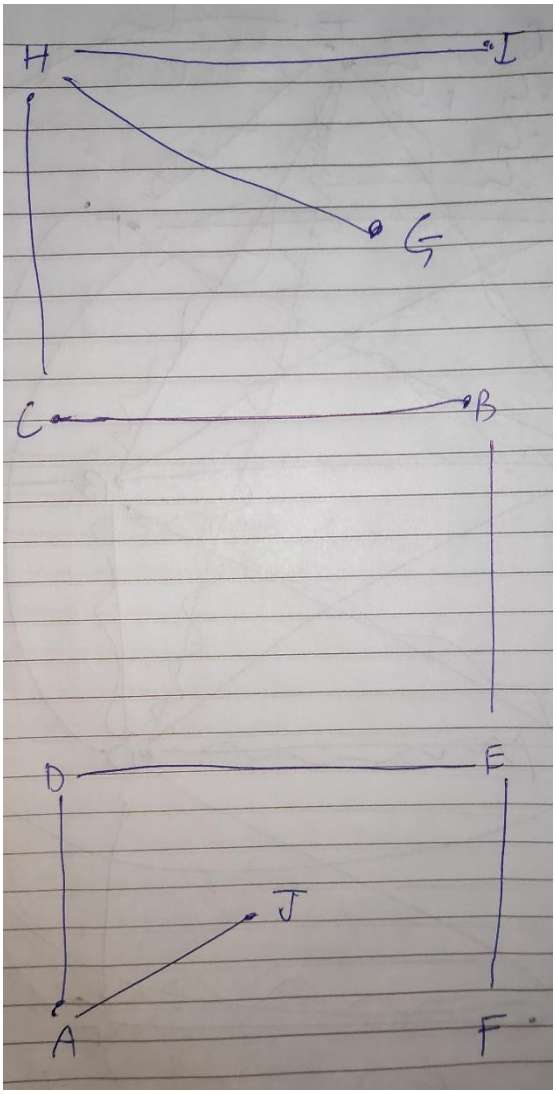
H=I
I=H
J=G

2
A=B
B=A
C=E
D=D
E=C
F=I
G=H
H=G
I=F
J=J

3

A=A
B=E
C=D
D=C
E=B
F=J
G=I
H=H
I=G
J=F

c)



Question 5

a)

$\{(x, y) \in R^2 \mid y = x^3 + 2\}$ is uncountably infinite.

$\forall x \in R, \exists y = x^3 + 2$. The function must be a bijective because the function is both surjective and injective as it passes the horizontal line test. Therefore ,
$\{(x, y) \in R^2 \mid y = x^3 + 2\} \cap [(0, 1) \times R] \sim (0, 1)$, so it must be uncountably infinite. We know that R is uncountably infinite so therefore the set must be uncountably infinite as there exists a bijection between R and the set.

b)

The set is countably infinite.

We know from the fundamental rule of algebra that a real polynomial equation would have at most the highest degree number of roots. Thus, meaning that this polynomial has at most three distinct solutions. Thus, each solution yielding up to three solutions. However, we have a set of polynomials bounded by integer coefficients. We get3 Z^4 possible quadruples $(a_0,\ a_1,\ a_2,\ a_3)$, where $a_0,\ a_1,\ a_2,\ a_3 \in$ Z, which is countably infinite. So, we get the cartesian product of finite and countably infinite which is countably infinite.

c)

We are restricted to N, between 0 and 15 as m and n must sum to 15 and they cannot exceed 15. We can get a total of 15 possible combinations of (m,n). The possible combinations on (m, n) are {(0, 15) ..... (15, 0)}. L must be finite with a total of 15 possible strings in L.

d)

It is countably infinite. It contains 0, the empty string (Ɛ), 1* and 01*.

We can put 1* and 01* in 1-1 correspondence with N as shown below. The amount of 1's (1*) is put into 1-1 correspondence with the odd numbers, and 0 followed by the number of 1's (01*) is put into 1-1 correspondence with the even numbers. For the odd numbers, we can calculate the amount of 1's by doing the following $1^{\frac{k+1}{2}}$, and for 0 followed by the amount of 1's (from the even numbers) can be calculated as following $01^{\frac{k}{2}}$, where the exponent means the amount of 1's present, and k is the number in N that maps to that particular string. For example, if we wanted to know what 7 maps to, we check: $1^{\frac{7+1}{2}} = 1^{\frac{8}{2}} = 1^4 = 1111$, and from the table below, it's correct.

N	0	1	2	3	4	5	6	7	8
$\tilde{L}$	0	1	01	11	011	111	0111	1111	01111

e)

Uncountably infinite. From lecture we saw that L(B) was empty, hence in DFA, there are no accepting states. There are no restrictions on E_{DFA} apart from the fact that it cannot have any accepting states. E_{DFA} is simply the set of all DFA with no accepting states. $E_{DFA} = \{<B> \mid B \text{ is a DFA and } L(B) = \emptyset\} \cap [(0, 1) \times R] \sim (0, 1)$. It must therefore be uncountably infinite as there are uncountably infinite ways to design a DFA with no accepting states.

<u>Question 6</u>

a)

$\tilde{L} = \{\ 0^m 1^{m-2} \mid m \in N,\ m \geq 2\}$. Assume $\tilde{L}$ has pumping length p where all string greater than or equal to length p can be pumped. We need to show $w = xy^iz$, for $i \geq 0$, and that $w \in L$, and meets following conditions.

1. $|y| \geq 1$
2. $|xy| \leq p$

Assume pumping length is 4. Let $w = 0001 \in L$ for $m = 3$. Try to decompose it into some xy^iz and pump it to see if greater values of i will put it in L. Let $x = 0$, Let $y = 00$ and let $z = 1$. With $i = 2$, w'' =000001 which is not in L. Try again Let $x = 00$, Let $y = 0$ and let $z = 1$. With $i = 2$, w'' =00001 which is not in L. Try again Let $x = 00$, Let $y = 01$ and let $z =$ empty. With $i = 2$, w'' =000101 which is not in L. Try again Let $x = 000$, Let $y = 1$ and let $z =$ empty. With $i = 2$, w'' =00011 which is not in L. We have exhausted all possible decompositions of the word 0001 and hence the language cannot be pumped and is therefore not regular.

b)

1) If the first cell is 0, then delete and move the pointer right. Otherwise, REJECT.
2) If the current cell is 0 delete it, then move the pointer right. Otherwise REJECT.
3) If the current cell is 0, go to step 4. If the current cell is blank, then ACCEPT. If the current cell is 1, then REJECT.
4) If the current cell is 0, delete it and move right to the first 1.
5) If no 1 is found, REJECT, otherwise change 1 to x.
6) Move to the leftmost (non-blank). If the current cell is 0, go to step 4. If the current cell is 1, REJECT. If the current cell is x, go to step 7.
7) Move right until something other than x or a blank is found. If the current cell is a 0 or a 1, then REJECT. If empty, ACCEPT.

c)

No. A counter example could be a language of all strings over a certain alphabet as it just accepts any string which itself is Turing decidable, but it contains non-Turing decidable language sublanguages which cannot be recognised by a Turing machine. Example: The English language is a subset of all strings formed by the Roman alphabet and cannot be Turing decidable.

d)

Let M be a Turing machine that recognises L_1 M' be a Turing machine that recognises L_2. $L_1, L_2 \subset A^*$.

In lectures it was shown that A^* is countably infinite so A^* has an enumeration, $A^* = \{w_1, w_2, w_3, w_4 \ldots\}$.

//essentially copy pasted from the lecture notes and adapted.

E = ignore the input

1) repeat the following for i = 1, 2, 3…
2) Run M for i steps on each input $w_1, w_2 \ldots w_i$
3) Run M' for i steps on each input $w_1, w_2 \ldots w_i$ on computations accepted by M otherwise go to step 1.

4) If any of the computations accepted in M' are found in M print out the corresponding w_j.

Every String accepted by M and M' will eventually appear on the list of E, and once it does it will appear infinitely many times because M and M' run from the beginning of each string for each repetition of step 1.