Module Title: Informatics 2A
Exam Diet (Dec/April/Aug): Dec 2010
Brief notes on answers:

PART A

1. d - because the string is in the language and its length depends on k
2. c - because the strings in the language comprise two copies of a string and this matches up the corresponding characters in the two strings.
3. c - the states count the number of bs encountered in mod 3 arithmetic
4. d - because the number of bs is divisible by three
5. a - because the string contains an odd number of as and so cannot be a member of R_2.
6. d - because it allows more than one parse tree for some strings in the language.
7. c - we can describe the language using a context-sensitive grammar but it is not CF
8. c - results in $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$
9. b - because these are the only letters that can begin a string derived from A and A cannot derive the empty string.
10. a - because the machine will not accept a b if the stack is empty.
11. b. Virtually all languages (arguably) have open classes of nouns and verbs, and very many have them for adjectives and adverbs. But prepositions, determiners, particles etc. are quite language-specific. In e, *may* is a hint; another example (mentioned in lectures) is *can*.
12. d, because training has to fill out a two-dimensional array of frequencies.
13. b (hymnic). A vowel is required before the suffix (in an attempt to ensure a stem of at least one syllable), but y is omitted.
14. c.
15. c.
16. b. LL(1) parsing can't be used e.g. if the attempt to build the LL(1) parse table gives rise to clashes.
17. d (owls hunt mice). The probabilities for a–e are respectively 0.028, 0.018, 0.03, 0.036, 0.0336.
18. e. Not d, because the sentence allows that X might take part but be unhappy because Y doesn't.
19. c. This one means *Everybody is loved by somebody.* The others all reduce to a.
20. c. The complement of the halting set is not r.e..

PART B

1. (a) The equations derivable from the machine are:

$$\begin{aligned} R_1 &= iR_2 + rR_1 + \varepsilon \\ R_2 &= dR_3 + fR_4 \\ R_3 &= sR_1 \\ R_4 &= rR_1 \end{aligned}$$

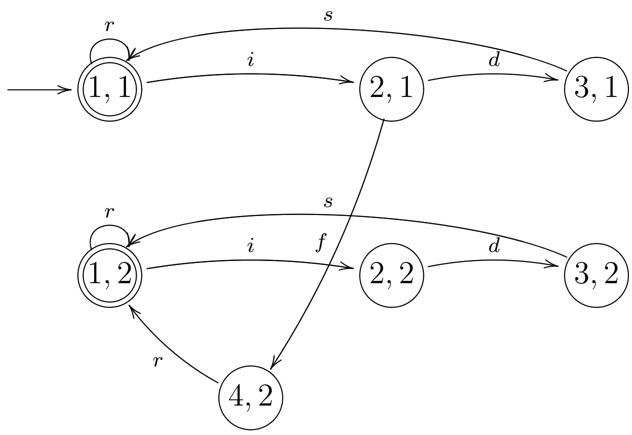
Allocate three marks for the equations.

Solving the equations:

$$\begin{aligned} R_2 &= fR_4 + dR_2 \\ R_2 &= frR_1 + dsR_1 \\ R_1 &= iR_2 + rR_1 + \varepsilon \\ R_1 &= i(fr + ds)R_1 + rR_1 + \varepsilon \\ R_1 &= (i(fr + ds) + r)R_1 + \varepsilon \\ R_1 &= (i(fr + ds) + r)* \end{aligned}$$

Allocate three marks for solving the equations.

(b) The intersection construction yields the machine:



Allocate 2 marks for getting the stateset correct, including the final state. Allocate a further two marks for the non-f transitions and one mark for the f transition.

(c) In this section we use the subset construction to construct the new machine. The stateset of the new machine comprises subsets of the states of the non-deterministic machine. The new transition function is:

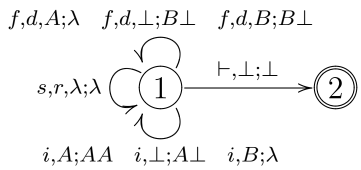
State	i	d	s	r	f
1	2				
2		3			1,4
3			1		
1, 4	2			1	

Award two marks for using subsets of the stateset and a further 4 for the transition function.

(d) - I disagree with the designer. *Award one mark*
  - Use the pumping lemma for regular languages to demonstrate the designer's claim is false:
    - Consider the string $x = i^{2k}d^k f^k \in L_4$.
    - By the pumping lemma there are strings $uvw = x$ with the length of uv less than k and $v \neq \varepsilon$ such that $uw \in L_4$
    - uv must be a prefix of i^2k
    - Since v is not the empty string, the string $i^{2k-|v|}d^k f^k$ must be in the language by the PL - it is not a member - so contradiction..

Award up to three marks for something that captures this line of argument (it need not be formal).

(e) The PDA is (state 1 is the initial state, I have omitted the in arrow on for legibility, the notation $s, c, f, \lambda; \lambda$ stands for three transitions $s, \lambda; \lambda, \ldots$:



Allocate 1 mark for getting the PDA notation correct, two marks for the i and f, d transitions and a further one mark for the other transitions.

2. (a) The parsing chart is as follows:

0	Det			NP	NP
1		DAdv	AP	Nom	Nom
2			Adj, AP	Nom	Nom
3				N, Nom, NP, Adj, AP	Nom
4					N, Nom, NP
	a	very	heavy	orange	book

There is only one parse for the whole noun phrase; its tree is obvious.

(b) The result is a graph with 6 nodes (0–5), 9 arcs and 20 dotted rule labels:

(0,1) a →. , Det →a. , NP →Det.Nom
(1,2) very →. , Adv →very . , AP →Adv.Adj
(2,3) heavy →. , Adj →heavy .
(1,3) AP →AdvD Adj . , Nom →AP.Nom
(3,4) orange →. , Adj →orange . , AP →Adj . , Nom →AP.Nom
(4,5) book →. , N →book . , Nom →N .
(3,5) Nom →AP Nom .
(1,5) Nom →AP Nom .
(0,5) NP →Det Nom .

The parsing strategy in question was covered in lectures as one among several, but I wouldn't expect total recall of the details of all of these, so an extended reminder seems appropriate here. Inessential variations in notation are acceptable.

(c) The LL(1) parse table is as follows:

	a	very	heavy	orange	book	$
NP	Det Nom					
Nom		AP Nom	AP Nom	AP Nom	N	
AP		AdvD Adj	Adj	Adj		
N					book	
Adj			heavy	orange		
Det	a					
Adv		very				

(d) We will encounter a clash when we try to build the parse table. The right hand sides N and AP Nom will compete for the entry (Nom, orange). (1 mark for saying the problem arises in building the table; 2 marks for the details.)

3. This question may take a while to assimilate but the amount of writing required for a correct solution is very modest. It is intended to appeal to more conceptually minded students. Part (e) is somewhat challenging and should serve as a differentiator for strong students.

(a) Very easy. 1 mark each.

(b) The following rules suffice:

$$\begin{array}{rcl} \text{NP} & \rightarrow & \text{a CN RelO} \\ \text{RelO} & \rightarrow & \epsilon \ \mid \text{Rel} \\ \text{Rel} & \rightarrow & \text{who VP} \ \mid \text{Name TV} \\ & & \mid \text{Name DV Name} \end{array}$$

1 mark per rule, roughly.

(c) The most obvious solution is as follows:

$$\begin{array}{llll} \mathsf{TV} & \rightarrow & \text{saw} & \{\ \lambda y.\lambda x.\ \text{saw}(x,y)\ \} \\ \mathsf{DV} & \rightarrow & \text{sent} & \{\ \lambda y.\lambda z.\lambda x.\ \text{sent}(x,y,z)\ \} \\ \mathsf{VP} & \rightarrow & \mathsf{TV}\ \mathsf{Name} & \{\ \mathsf{TV}.\text{Sem}\ (\mathsf{Name}.\text{Sem})\ \} \\ \mathsf{VP} & \rightarrow & \mathsf{DV}\ \mathsf{Name}_1\ \text{to}\ \mathsf{Name}_2 & \{\ \mathsf{DV}.\text{Sem}\ (\mathsf{Name}_1.\text{Sem})(\mathsf{Name}_2.\text{Sem})\ \} \end{array}$$

Note especially the order of the lambda abstractions. 1 mark per clause.

(d) E.g. for 'John introduced Mary to Susan', we have

$$\begin{array}{ll} & ((\lambda yzx.\ \text{introduced}(x,y,z))(\text{mary})(\text{susan}))(\text{john}) \\ \rightarrow & ((\lambda zx.\ \text{introduced}(\text{mary},y,z))(\text{susan}))(\text{john}) \\ \rightarrow & (\lambda x.\ \text{introduced}(x,\text{mary},\text{susan}))(\text{john}) \\ \rightarrow & \text{introduced}\ (\text{john},\text{mary},\text{susan}) \end{array}$$

2 marks for the raw expression, 2 for the reduction.

(e) For the rules given in (b) above, the following solution is probably the simplest:

$$\begin{array}{rcll} \text{NP} & \rightarrow & \text{a CN RelO} & \{\ \lambda x.\ \text{CN.Sem}(x) \wedge \text{RelO.Sem}(x)\ \} \\ \text{CN} & \rightarrow & \text{man} & \{\ \lambda x.\ \text{man}(x)\ \}, \quad \text{etc.} \\ \text{RelO} & \rightarrow & \epsilon & \{\ \lambda x.\ \text{TRUE}\ \} \\ \text{RelO} & \rightarrow & \text{Rel} & \{\ \text{Rel.Sem}\ \} \\ \text{Rel} & \rightarrow & \text{who VP} & \{\ \text{VP.Sem}\ \} \\ \text{Rel} & \rightarrow & \text{whom Name TV} & \{\ \lambda y.\ \text{TV.Sem}\ (y)\ (\text{Name.Sem})\ \} \\ \text{Rel} & \rightarrow & \text{to whom}\ \mathsf{Name}_1\ \text{DV}\ \mathsf{Name}_2 & \ldots \end{array}$$

$$\{\ \lambda z.\ \text{DV.Sem}(\mathsf{Name}_1.\text{Sem})(z)(\mathsf{Name}_2.\text{Sem})\}$$

Roughly 1 mark per rule. Quite difficult, so mark fairly generously.