

Examen de compiladorsI
Segona convocatòria 9 de juliol de 2007

Nom:

Cognoms:

Et presentes a la part de problemes? Si No

Respostes al test																	
1	a	b	c	d	e	11	a	b	c	d	e	21	a	b	c	d	e
2	a	b	c	d	e	12	a	b	c	d	e	22	a	b	c	d	e
3	a	b	c	d	e	13	a	b	c	d	e	23	a	b	c	d	e
4	a	b	c	d	e	14	a	b	c	d	e	24	a	b	c	d	e
5	a	b	c	d	e	15	a	b	c	d	e	25	a	b	c	d	e
6	a	b	c	d	e	16	a	b	c	d	e	26	a	b	c	d	e
7	a	b	c	d	e	17	a	b	c	d	e	27	a	b	c	d	e
8	a	b	c	d	e	18	a	b	c	d	e	28	a	b	c	d	e
9	a	b	c	d	e	19	a	b	c	d	e	29	a	b	c	d	e
10	a	b	c	d	e	20	a	b	c	d	e	30	a	b	c	d	e

Nota: Només una resposta és vàlida. En cas d'haver més d'una resposta vàlida i una tercera que les englobi nomes es considerarà correcta la que les engloba. Cada resposta correcta suma 1 punt, cada resposta incorrecta resta 0.25.

1.- Un compilador és un programa...

- a)...de poca complexitat que manega poc volum de dades.
- b)...de poca complexitat que manega molt volum de dades.
- c)...de molta complexitat que manega poc volum de dades.
- d)...de molta complexitat que manega molt volum de dades.
- e)Cap de les anteriors.

2.- Un scanner calcula els següents atributs d'un token:

- a) L'adreça i el desplaçament dels identificadors.
- b) El nom dels identificadors i el valor numèric dels números.
- c) La posició d'inici de les funcions.
- d) Totes les anteriors.
- e) Cap de les anteriors.

3.- Un scanner que es defineixi per un llenguatge amb paraules reservades i identificadors...

- a)s'implementa utilitzant un autòmat amb estats independents per cada símbol.
- b)s'implementa utilitzant un autòmat amb estats comuns pels símbols amb prefixes comuns.
- c)s'implementa sense utilitzar un autòmat.
- d)Totes les anteriors.
- e)Cap de les anteriors.

4.- Un analitzador lexicogràfic ens pot permetre...

- a)acceptar seqüències correctes de tokens, segons una gramàtica.
- b)acceptar els comentaris d'un programa.
- c)acceptar els tokens vàlids per aquell llenguatge.
- d)la b) i la c) són certes.
- e)Cap de les anteriors.

5.- Per implementar un Scanner fem servir:

- a) Un AFND perquè crearem una taula de transicions més curta.
- b) Un AFND perquè analitzarem la seqüència d'entrada més ràpidament.
- c) Un AFD perquè crearem una taula de transicions més curta.
- d) Un AFD perquè analitzarem la seqüència d'entrada més ràpidament.
- e) Cap de les anteriors.

6.-Per definir números enters utilitzant una expressió regular

- a) afegim el seu signe a la definició.
- b) no afegim el seu signe a la definició.
- c) si volem definir-los com un token, que formi part d'una expressió matemàtica de sumes i restes amb el menys unari, afegim el seu signe a la definició.
- d) Si volem definir-los com un token, que formi part d'una expressió matemàtica de sumes i restes amb el menys unari, no afegim el seu signe a la definició.
- e) Cap de les anteriors.

7.- La raó d'utilitzar una gramàtica LL(1) per definir un llenguatge de programació...

- a) és perquè ens permet tenir identificadors i crides a funció en el llenguatge.
- b) és perquè si no fos LL(1) seria ambigua.
- c) és perquè només podem definir llenguatges de programació utilitzant aquest tipus de gramàtiques.
- d) Totes les anteriors.
- e) Cap de les anteriors.

8.-Els analitzadors sintàctics top-down...

- a)serveixen per analitzar gramàtiques LL(1).
- b)serveixen per analitzar gramàtiques LR(k).
- c)serveixen per analitzar gramàtiques ambigües.
- d)Totes les anteriors.
- e)Cap de les anteriors.

9.- Un arbre sintàctic...

- a)representa l'estructura de les dades definides a un programa.
- b)representa l'estructura de l'ordre d'execució de les funcions i procediments d'un programa.
- c)representa l'estructura d'un programa, reflectint l'ordre de precedència de les operacions.
- d)representa l'estructura de la seqüència de crides a funcions i procediments que s'executaran en temps d'execució, definides a un programa.
- e)Cap de les anteriors.

10.- Les gramàtiques LL(1)...

- a) no tenen produccions λ .
- b) no tenen repeticions.
- c) no son BNF.
- d) Totes les anteriors.
- e) Cap de les anteriors.

11.-Si tenim un símbol no terminal que pot derivar en λ en una o més derivacions, la taula de prediccions, per aquell no terminal,

- a) té una entrada vàlida per cada un dels primers del no terminal.
- b) té una entrada vàlida per λ .
- c) a) i a més té una entrada vàlida per cada un dels següents del no terminal.
- d) b) i a més té una entrada vàlida per cada un dels següents del no terminal.
- e) Cap de les anteriors.

12.- Per generar un parser a partir de la descripció d'una gramàtica en BNF creem:

- a) Una funció per cada terminal T, un if per cada opcional [α] i un for per cada repetició de zero o més vegades una seqüència $\{\alpha\}$.
- b) Un funció per cada terminal T, un if per cada opcional [α] i un while per cada repetició de zero o més vegades una seqüència $\{\alpha\}$.
- c) Una funció per cada no-terminal $\langle n \rangle$, un if per cada opcional [α] i un for per cada repetició de zero o més vegades una seqüència $\{\alpha\}$.
- d) Una funció per cada no-terminal $\langle n \rangle$, un if per cada opcional [α] i un while per cada repetició de zero o més vegades una seqüència $\{\alpha\}$.
- e) Cap de les anteriors.

13.- Per corregir errors sintàctics en els analitzadors sintàctics descendents podem suposar...

- i.* es suposa que el codi erroni és codi addicional
- ii.* es suposa que falta codi
- k.* es salten símbols fins trobar un de correcte
- kk.* es salten símbols fins trobar un de correcte, com a símbol següent al no terminal que s'estava analitzant.

- a) *i.* i llavors es fa *k.*, o *ii.* i llavors es fa *k.*.
- b) *i.* i llavors es fa *k.*, o *ii.* i llavors es fa *kk.*
- c) *i.* i llavors es fa *kk.*, o *ii.* i llavors es fa *kk.*
- d) *i.* i llavors es fa *kk.*, o *ii.* i llavors es fa *k.*.
- e) Cap de les anteriors.

14.- El conjunt d'anul·lables d'una gramàtica LL(1) es defineix com:

- a) els terminals que deriven en lambda en un o més passos.
- b) els no-terminals que deriven en lambda en un o més passos.
- c) els terminals que deriven en lambda en un pas.
- d) els no-terminals que deriven en lambda en un pas.
- e) Cap de les anteriors.

15.- Els atributs dels símbols terminals i no-terminals es guarden a

- a) Els dels terminals a la TS i els del no-terminals a variables locals o paràmetres.
- b) Els dels terminals a la TS i els del no-terminals a la TS.
- c) Els dels terminals a variables locals o paràmetres i els del no-terminals a variables locals o paràmetres.
- d) Els dels terminals a variables locals o paràmetres i els del no-terminals a la TS.
- e) Cap de les anteriors.

16.- Quines comprovacions semàntiques s'han de fer en la regla BNF

$\langle \text{inst} \rangle ::= \text{if } \langle \text{exp} \rangle \text{ then } \langle \text{inst} \rangle [\text{else } \langle \text{inst} \rangle]$ de PASCAL?

- a) Verificar que el tipus de $\langle \text{exp} \rangle$ sigui un booleà
- b) a) i que les variables estiguin declarades
- c) b) i que las funcions estiguin declarades
- d) c) i que els tipus de les variables i funcions siguin coherents amb el seu ús
- e) Cap de les anteriors.

17.- Quina informació ha de guardar, en l'àmbit global, un compilador de C d'una funció?

- a) el seu nom, el tipus del valor de retorn i el tipus de cada argument.
- b) el seu nom, el tipus del valor de retorn, el tipus i nom de cada argument.
- c) el seu nom, el tipus del valor de retorn, el tipus i nom de cada argument i variable local.
- d) el seu nom, el tipus del valor de retorn, el tipus de cada argument i variable local.
- e) Cap de les anteriors.

18.- En el moment d'eliminar un símbol de la taula de símbols

- a) S'han d'eliminar tots els atributs associats al símbol exceptuant els compartits.
- b) S'han d'eliminar tots els atributs associats al símbol.
- c) S'han d'eliminar tots els atributs associats al símbol exceptuant els heretats.
- d) S'han d'eliminar tots els atributs associats al símbol exceptuant els sintetitzats.
- e) Cap de les anteriors.

19.- L'anàlisi semàntica d'expressions d'accés (ej. $a[i]^{\wedge}.camp$) suposa

- a) Recórrer l'estructura que representa el tipus de la variable.
- b) Fer la comprovació de que el tipus és un array i l'expressió de l'índex és un sencer quan trobem els claudàtors.
- c) Fer la comprovació de que el tipus és una estructura i que el camp pertany a aquesta quan trobem ".camp" i fer la comprovació de que el tipus és un apuntador quan trobem " \wedge ".
- d) Totes les anteriors
- e) Cap de les anteriors.

20.- De què depèn la mida d'una estructura de C?

- a) La mida del seus camps
- b) L'ordre del seus camps
- c) L'alineació
- d) Totes les anteriors
- e) Cap de les anteriors.

21.- De què depèn la mida d'una unió de C?

- a) La mida del seus camps.
- b) L'ordre del seus camps.
- c) L'alineació
- d) Totes les anteriors.
- e) Cap de les anteriors.

22.- Quantes multiplicacions necessita un accés a un array?

- a) Cap.
- b) Tantes com dimensions menys una.
- c) Tantes com dimensions.
- d) Tantes com dimensions mes una.
- e) Cap de les anteriors.

23.- Quina mida té en bytes el bloc d'activació de $\text{int } f(\text{int } a) \{ \text{int } c; \dots \}$ en un processador de 32 bits?

- a) 8
- b) 12
- c) 16
- d) 20
- e) Cap de les anteriors.

24.- ¿Per què una crida a una funció C pot ser més ràpida que una crida a una funció PASCAL equivalent?

- a) Pel display.
- b) Perquè la funció C retorna el resultat per un registre.
- c) Per la forma de passar els arguments a la funció C.
- d) Les funcions en PASCAL són igual de ràpides que les funcions en C.
- e) Cap de les anteriors.

25.- Quan es calcula el Display?

- a) La seva mida i el seu contingut el calcula el codi que realitza la trucada en temps d'execució.
- b) La seva mida i el seu contingut es calcula en temps de compilació
- c) La seva mida es calcula en temps de compilació i el seu contingut el calcula el codi que realitza la trucada en temps d'execució.
- d) La seva mida i el seu contingut es calcula en temps d'execució.
- e) Cap de les anteriors.

26.- Quina de les següents afirmacions és certa?

- a) L'enllaç estàtic del bloc d'activació d'una funció en execució apunta al bloc d'activació de la funció que l'ha trucat.
- b) L'enllaç dinàmic del bloc d'activació d'una funció en execució apunta al bloc d'activació de la funció on s'ha definit.
- c) El display s'utilitza per accedir a les variables locals de la funció en execució.
- d) La mida del display es calcula en temps d'execució.
- e) Cap de les anteriors.

27.- Durant l'execució d'un programa, als blocs d'activació de les funcions o procediments

- a) Pot haver més d'un bloc d'activació de funció incomplert.
- b) Hi ha com a màxim un bloc d'activació incomplert d'un procediment.
- c) Hi ha com a màxim un bloc d'activació incomplert.
- d) a) i b)
- e) Cap de les anteriors.

28.- Quants salts condicionals i incondicionals es necessiten en la generació de codi de la instruccions if (condició) instrucció else instrucció?

- a) 1
- b) 2
- c) 3
- d) 4
- e) Cap de les anteriors.

29.- Quantes sumes es realitzaran en temps d'execució per accedir a $v.camp1^{.}camp2$ on camp1 i camp2 són el segon i tercer camp d'una estructura.

- a) 1
- b) 2
- c) Una si v és variable global i tres si v és variable local.
- d) Una si v és variable local i tres si v és variable global.
- e) Cap de les anteriors.

30.- Quan podem generar el codi corresponent a l'aparició d'una variable dintre d'una expressió?

- a) Just després de llegir el seu nom.
- b) Tindrem que esperar per veure si ve el "=" d'una assignació o no.
- c) Tindrem que esperar per veure si ve el ";" del final de l'expressió.
- d) No tenim que generar codi per les variables que hi han a les expressions ja que tenen un espai de memòria reservat en temps de compilació.
- e) Cap de les anteriors.

Examen de **PROBLEMES** de compiladorsI
Segona convocatòria 9 de juliol de 2007

Nom:

Cognoms:

Problema1:

Definir la regla gramatical <repetir-finsque> i totes les necessàries per definir-la, seguint la notació BNF i fent que la gramàtica resultant sigui LL(1). Tenint en compte que la regla repetir-finsque és de la següent forma:

- Comença per la paraula reservada Repetir
- Té instruccions a dintre
- Acaba per la paraula reservada Fins-que i a continuació té una condició.

i a més:

- La regla <instrucció> ja està definida.
- Les condicions s'han de definir i poden ser comparacions (>,<,=) de números i identificadors, i ANDS, ORS i el NOT entre elements lògics.
- Nota: fem distinció entre valors lògics i numèrics.

Problema2:

Afegir a la següent part de la gramàtica del llenguatge LS l'anàlisi semàntica i la generació de codi en CoSeL o pseudocodi semblant.

Rule <DecFun> ::= Function identificador <DecParams> : <tipus> <bloc>

Rule <DecParams> ::= "(" [<DecParametre> { , <DecParametre> }])"

Rule <DecParametre> ::= identificador : <tipus>