

Nom:

Cognoms:

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 només es considerarà correcta la que les engloba. Cada resposta correcta suma 1 punt, cada resposta incorrecta resta 0.25.

1.- Una empresa ha desenvolupat una nova consola per la qual vol que els programadors desenvolupin nous jocs. Quin seria el compilador més apropiat per desenvolupar els jocs per la consola pensant en la facilitat de distribució del compilador i la velocitat d'execució dels jocs?

- a) Un compilador de C o C++ a codi màquina de la consola escrit en C o C++.
- b) Un compilador de C o C++ a codi màquina de la consola escrit en Java.
- c) Un compilador de Java a codi màquina de la consola escrit en Java.
- d) Un compilador de Java a codi màquina de la consola escrit en C o C++.
- e) Cap de les anteriors.

2.- Un analitzador lexicogràfic...

- a) ... determina si una seqüència de símbols d'entrada és correcta sense acceptar separadors, ni final de fitxer, ni símbols no terminals.
- b) ... determina si una seqüència de símbols d'entrada és correcta acceptant també els separadors però no el final de fitxer, ni els símbols no terminals.
- c) ... determina si una seqüència de símbols d'entrada és correcta acceptant també els separadors i el final de fitxer.
- d) ... determina si una seqüència de símbols d'entrada és correcta acceptant també els separadors, el final de fitxer i els símbols no terminals.
- e) Cap de les anteriors.

3.- Els atributs dels tokens els calcula...

- a) ...l'analitzador lexicogràfic
- b) ...l'analitzador sintàctic
- c) ...l'analitzador semàntic
- d) ...el generador de codi
- e) Cap de les anteriors.

4.- Un analitzador lexicogràfic s'implementa...

- a) ...mitjançant una autòmat finit determinista.**
- b) ...mitjançant una taula de transicions.**
- c) ...directament sense considerar un autòmat determinista.**
- d) Totes les anteriors.**
- e) Cap de les anteriors.**

5.- Un analitzador lexicogràfic

- a) ...és més eficient, en quant a espai, si reconeix en una única expressió regular les paraules reservades i els identificadors.**
- b) ...és més eficient, en quant a temps d'execució, si reconeix en una única expressió regular les paraules reservades i els identificadors.**
- c) ...és igual d'eficient, en quant a espai, tant si reconeix en una única expressió regular les paraules reservades i els identificadors, com si els reconeix amb una expressió regular per cada paraula reservada i una altre pels identificadors.**
- d) ...és igual d'eficient, en quant a temps d'execució, tant si reconeix en una única expressió regular les paraules reservades i els identificadors, com si els reconeix amb una expressió regular per cada paraula reservada i una altre pels identificadors.**
- e) Cap de les anteriors.**

6.- Per reconèixer tokens amb prefixos comuns...

- a) ...hem de reconèixer cada un d'aquests tokens per separat.**
- b) ...els hem de reconèixer junts i llegir caràcters per endavant per determinar quin és.**
- c) ...els hem de reconèixer junts i en el moment que arribem a un estat final donar el token associat a ell com a resultat.**
- d) Totes les anteriors.**
- e) Cap de les anteriors.**

7.- Un compilador d'anàlisi descendent d'una passada funciona de la següent manera:

- a) L'analitzador lexicogràfic comença llegint el fitxer d'entrada i va passant tokens i separadors a l'analitzador sintàctic que els va processant.**
- b) L'analitzador lexicogràfic comença llegint el fitxer d'entrada i va passant tokens a l'analitzador sintàctic que els va processant.**
- c) L'analitzador sintàctic comença l'anàlisi al símbol inicial de la gramàtica i va demanant tokens i separadors a l'analitzador lexicogràfic que els va llegint del fitxer d'entrada.**
- d) L'analitzador sintàctic comença l'anàlisi al símbol inicial de la gramàtica i va demanant tokens a l'analitzador lexicogràfic que els va llegint del fitxer d'entrada.**
- e) Cap de les anteriors.**

8.- Les gramàtiques que utilitzem per analitzar llenguatges de programació són

- a) Gramàtiques lliures de context perquè es poden analitzar amb una complexitat lineal**
- b) Gramàtiques depenents del context perquè el problema de l'anàlisi dels llenguatges és intrínsecament dependent del context**
- c) Gramàtiques regulars perquè sempre analitzem els tokens d'esquerra a dreta.**
- d) Gramàtiques generals perquè siguin prou potents per definir els nostres llenguatges de programació.**
- e) Cap de les anteriors.**

9.- La regla gramatical que podem utilitzar a una gramàtica LL(1) per una operació, OP, associativa per l'esquerra és:

- a) $\langle \text{operació} \rangle ::= \langle \text{operand} \rangle \{ \text{OP} \langle \text{operand} \rangle \}$ perquè podem simular la seva associativitat per l'esquerra analitzant l'arbre sintàctic d'esquerra a dreta.
- b) $\langle \text{operació} \rangle ::= \langle \text{operand} \rangle [\text{OP} \langle \text{operand} \rangle]$ perquè el seu arbre ja representa l'associativitat per l'esquerra.
- c) $\langle \text{operació} \rangle ::= \langle \text{operand} \rangle \{ \text{OP} \langle \text{operació} \rangle \}$ perquè podem simular la seva associativitat per l'esquerra analitzant l'arbre sintàctic d'esquerra a dreta.
- d) $\langle \text{operació} \rangle ::= \langle \text{operand} \rangle [\text{OP} \langle \text{operació} \rangle]$ perquè el seu arbre ja representa l'associativitat per l'esquerra.
- e) Cap de les anteriors

10.- Una gramàtica

- a) És ambigua quan no és LL(1).
- b) Si és LL(1) no és ambigua.
- c) És ambigua quan necessitem llegir més d'un token per endavant per poder crear l'arbre sintàctic
- d) Totes les anteriors
- e) Cap de les anteriors

11.- Una gramàtica $G(T, N, I, P)$ és LL(1) quan

- a) $\text{Primers}(N_i) \cap \text{Primers}(N_j) = \emptyset \quad \forall N_i, N_j \in N, i \neq j$
- b) $\text{Següents}(N_i) \cap \text{Següents}(N_j) = \emptyset \quad \forall N_i, N_j \in N, i \neq j$
- c) $\text{Primers}(N_i \rightarrow \beta) \cap \text{Primers}(N_i \rightarrow \beta') = \emptyset \quad \forall P_i \rightarrow \beta, P_i \rightarrow \beta' \in P : \beta \neq \beta'$
- d) $\text{Següents}(N_i \rightarrow \beta) \cap \text{Següents}(N_i \rightarrow \beta') = \emptyset \quad \forall P_i \rightarrow \beta, P_i \rightarrow \beta' \in P : \beta \neq \beta'$
- e) Cap de les anteriors

12.- Una gramàtica

- a) LR(1) s'analitza amb un analitzador sintàctic que analitza la seqüència d'entrada d'esquerra a dreta.
- b) LL(1) s'analitza amb un analitzador sintàctic que analitza la seqüència d'entrada d'esquerra a dreta.
- c) LR(1) s'analitza amb un analitzador sintàctic que utilitza la tècnica d'agafar la derivació de més a la dreta.
- d) Totes les anteriors.
- e) Cap de les anteriors.

13.- Els atributs dels símbols terminals i no terminals generats per l'analitzador semàntic es guarden a

- a) Els dels terminals a la TS i els dels no-terminals a l'arbre sintàctic.
- b) Els dels no terminals a la TS i els dels terminals a l'arbre sintàctic.
- c) Els dels terminals a la TS i els dels no terminals a la TS i a l'arbre sintàctic.
- d) Els dels no terminals i els dels terminals a l'arbre sintàctic.
- e) Cap de les anteriors.

14.- Donada una gramàtica per un llenguatge amb àmbits aniuats

- a) La taula de símbols tindrà un tipus de cerca d'un símbol: global.
- b) La taula de símbols tindrà dos tipus de cerca d'un símbol: local i global.
- c) La taula de símbols tindrà tres tipus de cerca d'un símbol: local, amb display i global.
- d) La taula de símbols tindrà quatre tipus de cerca d'un símbol: local, amb enllaç estàtic, amb display i global.
- e) Cap de les anteriors.

15.- Per poder analitzar tipus de dades recursius com els dels apuntadors

- a) Declarem el tipus de dades abans de processar la seva descripció.
- b) Detectem les recursivitats no permeses en les declaracions de tipus de dades.
- c) Declarem el tipus de dades després de processar la seva descripció.
- d) La a) i la b).
- e) Cap de les anteriors.

16.- Considerant que estem analitzant un llenguatge que només contempla el pas de paràmetres per valor. Quina informació mantindrà l'entrada a la taula de símbols per una funció, després d'analitzar semànticament aquesta funció?

- a) A l'entrada necessitem guardar: el nom de la funció i el seu tipus de retorn.
- b) La a) i la llista ordenada dels tipus dels seus paràmetres.
- c) La b) i la llista ordenada dels noms dels seus paràmetres.
- d) La c) i la llista ordenada dels noms i tipus de les seves variables.
- e) Cap de les anteriors.

17.- A una taula de símbols amb definició de funcions aniuades podem tenir:

- a) com a màxim una entrada a la taula de símbols de tipus separador d'àmbit de funció.
- b) com a màxim tantes entrades com funcions tinguem al programa que estiguem analitzant.
- c) com a mínim una entrada per cada funció que haguem analitzat fins aquell moment.
- d) Totes les anteriors.
- e) Cap de les anteriors.

18.- Quan analitzem semànticament l'accés a un array

- a) hem de verificar que el número de dimensions que li posem no sigui superior a les que tenim definides.
- b) la a) i hem de verificar que el tipus de dades de l'expressió amb la que fem l'accés sigui de tipus enter.
- c) la b) i hem de verificar que el valor de l'expressió amb la que fem l'accés no excedeixi el límit de l'array.
- d) la c) i hem de verificar que el valor de l'expressió amb la que fem l'accés no sigui negatiu.
- e) Cap de les anteriors.

19.- Per a quines declaracions el compilador genera adreces simbòliques?

- a) Per a les declaracions de variables, funcions i procediments.
- b) Per a les declaracions de variables globals, funcions globals i procediments globals.
- c) Per a les declaracions de variables, funcions globals i procediments globals.
- d) Per a les declaracions de variables globals, funcions i procediments.
- e) Cap de les anteriors.

20.- Per què els llenguatges actuals guarden els elements dels arrays per files?

- a) Perquè l'accés als elements de l'array és més ràpid que si es guarden per columnes.
- b) Perquè per un array de dos dimensions es pot especificar només el valor del primer índex i el resultat és un array d'una dimensió (per la definició `int a[10][20]`; `a[i]` seria un array de 20 sencers).
- c) la a) i la b).
- d) Perquè els elements del arrays guardats per files estan millor alineats a memòria que els guardats per columnes.
- e) Cap de les anteriors.

21.- Segons el compilador de C, quina és la mida en bytes de la següent estructura

```
struct {char a; int b; char c;};
```

- a) 3 bytes si els camps no estan alineats.
- b) 9 bytes si els camps estan alineats.
- c) 12 bytes si els camps estan alineats.
- d) 16 bytes si els camps estan alineats.
- e) Cap de les anteriors.

22.- El bloc d'activació d'una funció C conté

- a) Els paràmetres de l'últim al primer, l'enllaç dinàmic, l'adreça de retorn, les variables locals.
- b) El valor de retorn, els paràmetres de l'últim al primer, l'enllaç dinàmic, l'adreça de retorn, les variables locals.
- c) Els paràmetres del primer a l'últim, l'enllaç dinàmic, l'adreça de retorn, les variables locals.
- d) Els paràmetres de l'últim al primer, el display, l'enllaç dinàmic, l'adreça de retorn, les variables locals.
- e) Cap de les anteriors.

23.- Quina és la mida en bytes del display de la funció f , si f està definida dintre de la funció g i la funció g està definida dintre de la funció h ?

- a) 4
- b) 8
- c) 12
- d) 16
- e) Cap de les anteriors.

24.- Quin és el número màxim de valors que guardarem a la pila durant l'avaluació de l'expressió $a*b+c*d$?

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

- 25.-** Per què en LS es necessiten dues cues per guardar les instruccions quan es compilen les operacions aritmètiques?
- a) Per poder posar les conversions de tipus del primer operand abans del codi del segon operand.
 - b) Per poder posar les conversions de tipus del segon operand abans del codi de l'operació.
 - c) Per poder posar les instruccions de les operacions darrera dels seus operands.
 - d) la a) i la b).
 - e) Cap de les anteriors.
- 26.-** Dels següents conjunts d'instruccions, quines serien les que podríem utilitzar al trobar, a qualsevol lloc d'una expressió tant si porta assignació com si no, una variable local a la funció on ens trobem?
- a) IPopBVar, IPopDisplayVar, IPushBVar, IPushDisplayVar.
 - b) IPopBVar, IPopDisplayVar, IPushBVar, IPushDisplayVar, IStoreBVar, IStoreDisplayVar.
 - c) IPopBVar, IPushBVar, IStoreBVar.
 - d) IPopBVar, IPushBVar, IStoreBVar, IPushAddressBVar
 - e) Cap de les anteriors.
- 27.-** Quants salts genera la instrucció if amb else?
- a) un salt condicional.
 - b) un salt incondicional.
 - c) un salt condicional i un incondicional.
 - d) un salt condicional i dos incondicionals.
 - e) Cap de les anteriors.
- 28.-** A on es crea el display d'una funció?
- a) Al principi del codi de la funció.
 - b) Abans del codi dels paràmetres en la crida a la funció.
 - c) Abans de la instrucció ICall en la crida a la funció.
 - d) Després de la instrucció ICall en la crida a la funció.
 - e) Cap de les anteriors.
- 29.-** Quantes conversions de tipus generarà el compilador de LS per l'expressió $a+b*c+d$ si les variables s'han declarat com a:integer, b:integer, c:integer, i d:real?
- a) 1
 - b) 2
 - c) 3
 - d) 4
 - e) Cap de les anteriors.
- 30.-** Quans salts genera la instrucció while?
- a) un salt condicional.
 - b) un salt incondicional.
 - c) un salt condicional i un incondicional.
 - d) un salt condicional i dos incondicionals.
 - e) Cap de les anteriors.