

INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

COMPILADORES GURPO: 5CM4

GUIA 2

PROFESOR: TECLA PARRA ROBERTO

ALUMNO: LOPEZ PEREZ ALBERTO ANDREI

# Falso y verdadero

1.- En lenguaje C los parámetros formales son como variables locales que ya fueron inicializadas en el momento de la llamada a la función o procedimiento.

verdadero

2.-En lenguaje C las variables locales (no estáticas) se crean cuando se entra a una función y se destruyen cuando sale de la función.

verdadero

3.-En hoc los parámetros usados dentro de la definición de las funciones no tienen nombre. falso

4.-No es posible definir funciones recursivas en hoc. falso

5.-En hoc no hay variables locales. verdadero

6.- En hoc cuando una función termina su ejecución se saca su marco de la pila de llamadas.

verdadero

7.- En hoc los parámetros reales son listas de expresiones. verdadero

8.-En hoc el código que ejecuta la máquina virtual de pila está en prefijo (considere como se ejecuta una operación de suma).

falso

9.-En hoc el tipo de las variables es entero. falso

10.-en hoc los parámetros reales se meten a la pila. falso

# Opción múltiple

1.-Que es una macro en C?

1. un alias
2. algo que cuando el compilador (o preprocesador) encuentra lo sustituye con otra cosa
3. un nombre para un conjunto de instrucciones

2.-Que es una macro con parámetros en C?

1. un alias
2. una función
3. una macro a la que se le dan los parámetros dentro de paréntesis y separados por comas
4. un nombre para un conjunto de instrucciones

3.-En hoc2 que se usa como tabla de símbolos? a) un arreglo

1. una pila
2. una lista simplemente ligada
3. una unión

4.-En hoc3 que se usa como tabla de símbolos?

1. un arreglo
2. una pila
3. una lista simplemente ligada
4. una unión

5.-Cual es el tipo de las variables en hoc2?

a) int b) float c) double d)union

6.-Cual es el tipo de las variables en hoc3?

a) int b) float c) double d)union

7.-Cual es el tamaño máximo del nombre de las variables en hoc2? a) 1 b) 2 c) 3 d) el tamaño del arreglo sbuf menos uno

8.-Cual es el tamaño máximo del nombre de las variables en hoc3?

a) 1 b) 2 c) 3 d) el tamaño del arreglo sbuf menos uno

9.-En hoc2 donde se almacena el valor de una variable?

1. en la posición del arreglo que corresponde a esa variable
2. en la parte val de la unión de la entrada en la tabla de símbolos donde se guardó el nombre de la variable
3. en la RAM de la Máquina Virtual de Pila
4. en la Pila

10.-En hoc3 donde se almacena el valor de una variable?

1. en la posición del arreglo que corresponde a esa variable
2. en la parte val de la unión de la entrada en la tabla de símbolos donde se guardó el nombre de la variable
3. en la RAM de la Máquina Virtual de Pila
4. en la Pila

11.-Que es un apuntador?

1. una dirección de una localidad de memoria
2. una variable que almacena la dirección de una localidad de memoria
3. una variable

12.-Para qué sirve un apuntador?

a) para almacenar la dirección de una localidad de memoria b) para manejar las cosas de forma indirecta

c) para paso de parámetros por referencia

13.-Que introduce un apuntador simple? a) un nivel de indirección

1. dos niveles de indirección
2. tres niveles de indirección
3. cuatro niveles de indirección

14.-Que introduce un apuntador doble?

1. un nivel de indirección
2. dos niveles de indirección
3. tres niveles de indirección
4. cuatro niveles de indirección

15.-Que introduce un apuntador triple?

1. un nivel de indirección
2. dos niveles de indirección c) tres niveles de indirección

d) cuatro niveles de indirección

16.-Por que los apuntadores hacen muy poderoso a lenguaje C?

a) hacen los programas de C más fáciles de entender b) permiten manejar las cosas de forma flexible

c) permiten que haya apuntadores salvajes

17.-Que valor proporciona el nombre de un arreglo?

1. la dirección del arreglo
2. la dirección del primer elemento del arreglo
3. la dirección del último elemento del arreglo

18.-Que valor proporciona el nombre de una función?

1. la dirección de la función
2. la dirección de la primera instrucción de la función
3. la dirección de la última instrucción de la función 19.-Que es un apuntador a función?
4. una variable que almacena la dirección de una función
5. una variable que almacena la dirección de la primera instrucción de una función
6. una variable que almacena una función

20.-Para qué sirve un apuntador a función?

1. para almacenar la dirección de la primera instrucción de una función
2. para llamar a una función
3. para llamar de forma indirecta a una función

21.-Que es double (\*a)(double)?

1. un double
2. una apuntador a un double
3. una función cuyo tipo de retorno es double y que recibe como parámetro un double
4. una función cuyo tipo de retorno es apuntador a double y que recibe como parámetro un double
5. una apuntador a función cuyo tipo de retorno es double y que recibe como parámetro un double

22.-Que hace la función lookup?

1. crea una entrada de la tabla de símbolos, la llena a partir de los parámetros y la inserta al inicio
2. crea una entrada de la tabla de símbolos, la llena a partir de los parámetros y la inserta al final
3. verifica si la cadena que recibe como parámetro es el nombre de una entrada de la tabla de símbolos
4. guarda las constantes y los builtins en la tabla de símbolos

23.-Que hace la función install?

1. crea una entrada de la tabla de símbolos, la llena a partir de los parámetros y la inserta al inicio
2. crea una entrada de la tabla de símbolos, la llena a partir de los parámetros y la inserta al final
3. verifica si la cadena que recibe como parámetro es el nombre de una entrada de la tabla de símbolos
4. guarda las constantes y los builtins en la tabla de símbolos

24.-Que hace la función init?

1. crea una entrada de la tabla de símbolos, la llena a partir de los parámetros y la inserta al inicio
2. crea una entrada de la tabla de símbolos, la llena a partir de los parámetros y la inserta al final
3. verifica si la cadena que recibe como parámetro es el nombre de una entrada de la tabla de símbolos
4. guarda las constantes y los builtins en la tabla de símbolos

25.-Cuantas pilas hay en hoc1? a) 1 b) 2 c) 3 d) ninguna

26.-Cuantas pilas hay en hoc4?

a) 1 b) 2 c) 3 d) ninguna

27.-Cual es el tipo por default de los elementos en la pila de YACC?

a) int b ) float c) double c) union

28.-Cual es el tipo de los elementos en la pila de YACC en HOC1?

a) int b ) float c) double c) union

29.-Cual es el tipo de los elementos en la pila de YACC en HOC2?

a) int b ) float c) double c) union

30.-Cual es el tipo de los elementos en la pila de YACC en HOC3?

a) int b ) float c) double c) union

31.-Cual es el tipo de los elementos en la pila de YACC en HOC4?

a) int b ) float c) double c) union

32.-Que es void (\*p)(double)?

1. una apuntador a void
2. una función cuyo tipo de retorno es void y que no recibe parámetros
3. una función cuyo tipo de retorno es apuntador a void y que no recibe parámetros d) una apuntador a función cuyo tipo de retorno es void y que no recibe parámetros

33.-en la función constpush se usan?

a) la pila de yacc , la tabla de símbolos y la RAM de la Máquina Virtual de Pila (MVP) b) la pila de la MVP, la tabla de símbolos y la RAM de la MVP

1. la pila de la MVP y la tabla de símbolos
2. la tabla de símbolos y la RAM de la MVP

34.-Que hace la función initcode?

1. almacena el primer elemento del arreglo stack en stackp y el 1er elemento del arreglo prog en progp
2. almacena la dirección del primer elemento del arreglo stack en stackp y almacena la dirección del 1er elemento del arreglo prog en progp
3. guarda las constantes y los builtins en la tabla de símbolos
4. ejecuta una serie de instrucciones iniciado en la localidad de la RAM de la MVP cuya dirección se le pasa como parámetro y termina cuando llega a un STOP

35.-Que hace la función code?

1. almacena la instrucción que se le pasa como parámetro en la RAM de la MVP
2. almacena la instrucción que se le pasa como parámetro en la la tabla de símbolos
3. almacena la instrucción que se le pasa como parámetro en la pila de la MVP
4. ejecuta la instrucción que se le pasa como parámetro 36.-Que valor retorna la función code?
5. la dirección de la RAM de la MVP donde se almacenara la siguiente instrucción
6. una dirección de la RAM de la MVP c)

la dirección de la RAM de la MVP donde se guardó la instrucción que se le pasa como

parámetro

d) un apuntador a Inst

37.-En la función add se hace lo sig?

1. se toman 2 operandos de la RAM de la MVP, luego se suman y la suma se guarda en la RAM de la MVP
2. se toman 2 operandos de la RAM de la MVP, luego se suman y la suma se mete a la pila de la MVP
3. se sacan 2 operandos de la pila de la MVP, luego se suman y la suma se mete a la pila de la MVP
4. se sacan 2 operandos de la pila de la MVP, luego se suman y la suma se mete a la tabla de símbolos

38.-Como genera hoc4 el código?

1. como si estuviera en notación en notación prefija
2. como si estuviera en notación en notación infija c) como si estuviera en notación en notación postfija

39.-Que hace la función execute?

1. ejecuta una serie de instrucciones iniciado en la localidad de la RAM de la MVP cuya dirección se le pasa como parámetro y termina cuando llega a un STOP
2. ejecuta una serie de instrucciones iniciado en la primera localidad de la RAM de la MVP y termina cuando llega a un STOP
3. almacena la instrucción que se le pasa como parámetro en la RAM de la MVP
4. almacena la instrucción que se le pasa como parámetro en la pila de la MVP

40.-Cual es el tipo de los elementos en la pila de la MVP?

a) int b) double c) Symbol d) Datum e) Inst

41.-Cual es el tipo de los elementos en la RAM de la MVP?

a) int b) double c) Symbol d) Datum e) Inst

42.-Que almacena pc?

1. almacena la dirección de la localidad de memoria actual
2. almacena la dirección de la localidad de memoria donde está la siguiente instrucción a ejecutarse
3. almacena la dirección de la localidad de memoria de la RAM de la MVP donde se almacenara la siguiente instrucción

43.-que almacena progp?

1. almacena la dirección de la localidad de memoria actual
2. almacena la dirección de la localidad de memoria donde está la siguiente instrucción a ejecutarse
3. almacena la dirección de la localidad de memoria de la RAM de la MVP donde se

almacenara la siguiente instrucción

44.-Que función genera el código en hoc4

a) yylex() b) yyparse() c) code() d) execute() e) initcode()

45.- ¿Que almacena progbase?

1. almacena la dirección de la localidad de memoria actual
2. almacena la dirección de la localidad de memoria donde está la siguiente instrucción a ejecutarse
3. almacena la dirección de la localidad de memoria de la RAM de la MVP donde se almacenará la siguiente instrucción.
4. almacena la dirección de la localidad de memoria donde esta la dirección de la 1er instrucción del cuerpo de una función o un procedimiento.

46.- ¿Que se almacena en una macro de función en lenguaje C? a) El nombre de la función llamada, los parámetros

1. Los parámetros, dirección de retorno, las variables locales
2. Los parámetros, el contador de programa, las variables locales.

47.- ¿Que se almacena en la pila de llamadas?

a) símbolos gramaticales b) macros de función

1. resultados de evaluación de expresiones
2. tokens

48.-De lo mencionado abajo, ¿dónde no hay saltos del contador de programa? a) secuencia.

1. decisiones (if).
2. ciclos.
3. funciones y/o procedimientos.

49.-Que código genera hoc4 si el usuario escribe lo de abajo a) 5 + 9 \* 2

|  |
| --- |
| RAM |
| Constpush |
| 5 |
| Constpush |
| 9 |
| Constpush |
| 2 |
| Mul |
| Add |
| Print |
| Stop |

b) x+y

|  |
| --- |
| RAM |
| Varpush |
| X |
| Eval |
| Varpush |

|  |
| --- |
| Y |
| Eval |
| Add |
| Print |
| Stop |

c) (2+x)\*(y+5)

|  |
| --- |
| RAM |
| Constpush |
| 2 |
| Varpush |
| X |
| Eval |
| Add |
| Varpush |
| Y |
| Eval |
| Constpush |
| 5 |
| Add |
| Mul |
| Print |

1. aa = 10 \* sin( PI/4 )

|  |
| --- |
| RAM |
| Consts |
| PI |
| Constpush |
| 4 |
| Div |
| Bltin |
| Varpush |
| Sin |
| Constpush |
| 10 |
| Mul |
| Varpush |
| Aa |
| Assign |
| Pop |
| Stop |

1. bb + cc \* ( dd+ee\*100 )

|  |
| --- |
| Varpush |
| Bb |
| Eval |
| Varpush |
| Cc |
| Eval |
| Varpush |
| Dd |
| Eval |
| Varpush |
| Ee |
| Eval |
| Constpush |
| 100 |
| Mul |
| Add |
| Mul |
| Add |
| Print |
| Stop |