**Examen de Programación (Grados en Ingeniería Mecánica, Eléctrica, Electrónica y Química Industrial)**

**6 de Junio 2022**

**APELLIDO/S: NOMBRE:**

**DNI: GRUPO DOCENTE: GRUPO TRABAJO:**  **REPOSITORIO TI:**

**Instrucciones**:

Para seleccionar la respuesta (o respuestas) correcta(s), rodear la letra de la opción con un círculo **O**

Para anular una respuesta ya seleccionada tachar el círculo con una cruz **X**

Para volver a seleccionar de nuevo una respuesta anulada usar una flecha a la izq. de la opción **🡪**

El examen consta de 30 preguntas tipo test de respuestas múltiples

Cada pregunta admite una o varias opciones correctas (hay que seleccionar todas las opciones correctas)

Puntuación de cada pregunta: 8.0/30 (si se han seleccionado solamente todas las opciones correctas)

0.0 (en caso contrario)

* Un disco duro de un TeraByte (TB), ¿cuántos bits aproximadamente puede almacenar?

1. 1.000.000.000.000
2. 1.000.000.000.000.000
3. 4.000.000.000.000
4. 4.000.000.000.000.000
5. 8.000.000.000.000
6. 8.000.000.000.000.000
7. Ninguna de las anteriores

* Considerar la representación binaria sesgada (o desplazada) para los números enteros con signo, con un tamaño de representación de 6 dígitos. ¿Cuál sería la representación del número -21?

1. 110101
2. 101011
3. 101010
4. 100101
5. 010101
6. 001011
7. 001010
8. 000101
9. 100000
10. 011111
11. Ninguna de las anteriores

* En cuál de los siguientes registros internos de la Unidad de Control de un procesador central (CPU) se almacena la dirección de memoria central de la siguiente instrucción a ejecutar en dicho procesador:

1. IR (“Instruction Record” 🡪 Registro de Instrucción)
2. PC (“Program Counter” 🡪 Contador de Programa)
3. SP (“Stack Pointer” 🡪 Puntero a la cima de la Pila)
4. En cualquiera de los anteriores.
5. En ninguno de los anteriores

* ¿Cuál o cuáles de los siguientes identificadores son válidos en lenguaje C?

1. E+23
2. DOS PI
3. 30Junio
4. real
5. false
6. array
7. nota.primera
8. tipo\_vector
9. double
10. speed
11. Ninguno de los anteriores es válido

* En lenguaje C, ¿qué instrucción/instrucciones de las siguientes utilizaría para escribir en pantalla un número real de doble precisión almacenado en la variable **x**, en un campo de tamaño 10 a partir de la posición actual del cursor, justificado a la derecha y con 2 cifras decimales?
  1. printf("%10.2d", x);
  2. printf("%13.2d", x);
  3. printf("%10.2f", x);
  4. printf("%13.2f", x);
  5. printf("%10.2r", x);
  6. printf("%13.2r", x);
  7. printf("%10.2lr", x);
  8. printf("%13.2lr", x);
  9. printf("%-10.2lr", x);
  10. printf("%-13.2lr", x);
  11. Ninguna de las anteriores
* Dado el siguiente programa en lenguaje C, señale la afirmación o afirmaciones correctas:

|  |  |
| --- | --- |
| * 1. El programa lee por teclado dos números reales y luego imprime su diferencia en valor absoluto   2. El programa lee por teclado dos números reales y luego siempre imprime 0.00 de distancia   3. Error de compilación: falta la librería math.h   4. Error de compilación: falta un ; tras llave de cierre }   5. Error en tiempo de ejecución: a y b tendrían que haberse declarado como reales en doble precisión   6. Todas las afirmaciones anteriores son incorrectas | #include <stdio.h>  #include <stdlib.h>  int main(void){  float a,b;  printf("Introduzca posición 1: ");  scanf(" %f", &a);  printf("Introduzca posicion 2: ");  scanf(" %f",&b);  printf("Distancia: %.2f", abs(b-a));  return 0;  } |

* ¿Qué mensaje aparece en pantalla al ejecutar el siguiente programa en C?

|  |  |
| --- | --- |
| * 1. No forman un triangulo   2. Area = 3   3. Area = 3.1   4. Area = 3.11   5. Area = 0.00   6. (No aparece nada en pantalla)   7. Error de compilación   8. Ninguna de las anteriores | #include <stdio.h>  #include <math.h>  int main(){  double a=2.0,b=3.0,c=5.1,area,sp;  if((a>b+c)&&(b>a+c)&&(c>a+b))  printf("No forman un triangulo");  else{ sp=(a+b+c)/2.0;  area=sqrt((sp-a)\*(sp-b)\*(sp-c));  printf("Area = %.2f\n",area);  }  return 0;  } |

* ¿Qué mensaje se presenta en pantalla cuando se ejecuta el siguiente programa?

|  |  |
| --- | --- |
| * 1. El valor de x es 0   2. El valor de x es 1   3. El valor de x es 2   4. El valor de x es distinto de 0, 1 y 2   5. (No aparece nada en pantalla)   6. Error de compilación   7. Error de ejecución | #include <stdio.h>  int main(void) {  float x=2;  switch(x){  case 0: printf("El valor de x es 0");  break;  case 1: printf("El valor de x es 1");  break;  case 2: printf("El valor de x es 2");  break;  default: printf("El valor de x es ");  printf("distinto de 0, 1 y 2");  }  return 0;  } |

* Indicar el resultado del siguiente programa en C:

|  |  |
| --- | --- |
| * 1. Lee por teclado un número real en doble precisión en el intervalo [10,20] y lo escribe a continuación en pantalla con tres cifras decimales.   2. Lee por teclado un número real en doble precisión en el intervalo (10,20) y lo escribe a continuación en pantalla con tres cifras decimales.   3. Lee por teclado un número real y escribe a continuación: x=0.000   4. Error de compilación   5. Error de ejecución (bucle infinito) | #include <stdio.h>  int main(){  double x;  do{ printf("Numero [10,20]: ");  scanf(" %lf",&x);  }while((x>10)||(x<20));  printf("x=%.3f\n", x);  return 0;  } |

* Tras ejecutar el siguiente programa en C, ¿qué valores se muestran por pantalla?

|  |  |
| --- | --- |
| 1. 0 2. 11 3. 11-5-2-1 4. 11-34-17-52-26-13-40-20-10-5-16-8-4-2-1 5. 11-33-18-42-21-10-30-22-9-5-4-8-6-3-2-1 6. 11-22-44-66-55-33-10-20-40-30-15-5-3-1 7. (No aparece nada en pantalla) 8. Error de compilación 9. Error de ejecución (bucle infinito) | #include <stdio.h>  int main(){  int n=11;  printf("%d",n);  while(n!=1){  if(n%2) n=3\*n+1;  else n=n/2;  printf("-%d",n);  }  return 0;  } |

* ¿Qué mensaje se presenta en pantalla cuando se ejecuta el siguiente programa?

|  |  |
| --- | --- |
| * 1. El valor de x es 500   2. El valor de x es 525   3. El valor de x es 550   4. El valor de x es 575   5. El valor de x es 600   6. El valor de x es 1000   7. (No aparece nada en pantalla)   8. Error de compilación   9. Error de ejecución (bucle infinito) | #include <stdio.h>  int main(void) {  float x=0.0;  int i,j,k;  for(i=1;i<=10;i++)  for(j=1;j<=10;j++)  for(k=j;k<=10;k++)  x++;  printf("El valor de x es %.0f",x);  return 0;  } |

* ¿Cuál (o cuáles) de las siguientes afirmaciones es cierta en relación con la definición de funciones en C?

1. Un programa C consta de varias definiciones de funciones, que se pueden escribir en cualquier orden y tener cualquier nombre (siempre que se respeten las reglas de los identificadores de C).
2. Toda función devuelve por defecto a través de su identificador un resultado de tipo **int**, a no ser que se especifique bien un tipo de dato diferente o bien que no devuelve nada.
3. La función denominada **main** es la que inicia la ejecución del programa, y ésta puede contener instrucciones de llamada a otras funciones.
4. Es obligatorio que antes de las definiciones de las funciones (al principio del código fuente) se incluyan los prototipos de las funciones (primera línea de la definición de cada función acabada en ;).
5. La inclusión de la librería **stdio.h** permite construir programas que constan de más de una definición de función.
6. Todas las afirmaciones anteriores son incorrectas.

* ¿Cuál es el resultado que se muestra por pantalla tras ejecutar el siguiente programa?

|  |  |
| --- | --- |
| * 1. x=3 y=5   2. x=5 y=3   3. x=3.000000 y=5.000000   4. x=5.000000 y=3.000000   5. Error de sintaxis   6. Ninguna de las respuestas anteriores | #include <stdio.h>  void intercambiar(double \*x, double \*y);  int main(void){  double x=3, y=5;  intercambiar(&x,&y);  printf("x=%.0f y=%.0f",x,y);  return 0;  }  void intercambiar(double \*x, double \*y){  \*x=\*y-\*x;  \*y=\*y-\*x;  \*x=\*x+\*y;  } |

* Considerar el siguiente fragmento de código en lenguaje de programación C. ¿Cuál o cuáles de las siguientes instrucciones puede utilizar en la línea de puntos --------- de la función leer\_datos para que el compilador no genere ningún error ni aviso (“warning”)?

|  |  |
| --- | --- |
| 1. leer\_entero\_rango(0,&n,&m); 2. leer\_entero\_rango(0,n,\*m); 3. leer\_entero\_rango(0,&(n+m),&m); 4. leer\_entero\_rango(0,\*(n+m),m); 5. Ninguna de las anteriores | #include <stdio.h>  void leer\_datos(int \*n, int \*m);  void leer\_entero\_rango(int a,int b,int \*m);  int main(){  int n,m;  leer\_datos(&n,&m);  return(0);  }  void leer\_datos(int \*n, int \*m){  leer\_entero\_rango(0,100,n);  ----------  }  void leer\_entero\_rango(int a,int b,int \*m){  /\* Definición de función \*/  } |

* Dada la siguiente definición de función en C, indicar el resultado almacenado en la variable **x** tras la siguiente activación: x=leer\_entero\_intervalo(5,-5);

|  |  |
| --- | --- |
| 1. Un valor entero no nulo en intervalo [-5,5] 2. Un valor entero en intervalo [-5,5] 3. Un valor entero en intervalo (-5,5) 4. Un valor entero fuera del intervalo (-5,5) o bien el valor 0 5. Un valor entero no nulo cualquiera 6. Error de compilación 7. Error de ejecución: bucle infinito | int leer\_entero\_intervalo(int a, int b){  int x;  do{ printf("Numero [%d,%d]: ",a,b);  scanf(" %d",&x);  }while((x<a)&&(x>b)&&(!x));  return x;  } |

* Dado el siguiente fragmento de programa: ¿cuál ó cuáles de las activaciones de la función **f** podría escribirse en la línea de guiones (-----------) para que el programa compilase sin errores ni avisos (“warnings”)?

|  |  |
| --- | --- |
| 1. y=f(x,c,z); 2. y=f(&x,&c,&z); 3. y=f(x+3,c,z); 4. y=f(\*x,\*c,\*z); 5. Todas las anteriores son incorrectas. | #include <stdio.h>  int f(int \*x, char \*c, int \*z);  int main(){  int x=4, y, \*z;  char c='C';  z=&x;  -------------------  return(0);  }  int f(int \*x, char \*c, int \*z){  /\* Definición de función \*/  } |

* Indicar qué escribe en pantalla el siguiente programa al ejecutarlo:

|  |  |
| --- | --- |
| 1. verdadero 2. falso 3. 0 4. 1 5. 1 0 1 0 1 0 6. 0 1 0 1 0 1 7. Error de compilación 8. Error en tiempo de ejecución | #include <stdio.h>  int es\_par(int n);  int es\_impar(int n);  int main(){  printf(" %d" , es\_impar(6));  return 0;  }  int es\_par(int n){  if(n) return(es\_impar(n-1));  else return 1;  }  int es\_impar(int n){  if(n) return(es\_par(n-1));  else return 0;  } |

* Considere el siguiente prototipo de una función C: double f(double (\*f)(double x),double x);

¿Cuál o cuáles de las siguientes instrucciones son correctas?

1. printf(" Resultado= %f" ,f(sin,M\_PI));
2. printf(" Longitud= %f" ,f(atan,1.0));
3. printf(" Longitud= %f" ,f(pow,5.0,4.0));
4. printf(" Longitud= %f" ,f(cos+sin,M\_PI));
5. Ninguna de las anteriores

* ¿Cuál (o cuáles) de las siguientes instrucciones escribe en pantalla un número real situado en la posición i-ésima de un “array” unidimensional de reales de doble precisión en C de nombre **v**?
  1. printf(" %.1f", \*v+i);
  2. printf(" %.1f", \*(v+i));
  3. printf(" %.1f", v[i+1]);
  4. printf(" %.1f", &v[i]);
  5. Ninguna de las anteriores
* Considerar el siguiente programa en C, e indicar qué resultado(s) aparecería en pantalla tras su ejecución:

|  |  |
| --- | --- |
| * 1. 1 -3 -5 2 8 -7 4   2. 1 -3 -5 2 -5 -3 1   3. 4 -7 8 2 -5 -3 1   4. 1 1 1 1 1 1 1   5. 4 4 4 4 4 4 4   6. Error de sintaxis (falta el prototipo de la función **invertir**)   7. Ninguna de las anteriores | #include <stdio.h>  #define MAX 100  typedef int tipo\_vector[MAX];  void invertir(tipo\_vector v, int n){  int i,x;  for(i=0;i<n;++i){  x=v[i];  v[i]=v[n-i-1];  v[n-i-1]=x;  }  }  int main(){  tipo\_vector v={1,-3,-5,2,8,-7,4};  int n=7,i;  for(i=0;i<n;++i)  printf("%d ",v[i]);  return 0;  } |

* De las siguientes declaraciones de nuevos tipos de datos en C, indicar cuál o cuáles seleccionaría para representar la cantidad de masa vegetal en kg que hay en cada metro cúbico de un invernadero con forma rectangular y cubierta plana horizontal, cuyas dimensiones son 100 m de largo, 50 m de ancho y 5 m de alto:

1. typedef double tipo[99][49][4];
2. typedef double tipo[49][99][4];
3. typedef double tipo[4][49][99];
4. typedef double tipo[100][50][5];
5. typedef double tipo[50][100][5];
6. typedef double tipo[5][50][100];
7. typedef double tipo[101][51][6];
8. typedef double tipo[51][101][6];
9. typedef double tipo[6][51][101];
10. Ninguna de las anteriores

* ¿Cuál o cuáles de los siguientes fragmentos de código en C utilizaría para calcular la matriz traspuesta de una matriz dada?

|  |  |
| --- | --- |
| * 1. for(i=0;i<ca;++i)   for(j=0;j<fa;++j)  B[i][j]=A[j][i];   * 1. for(i=0;i<fa;++i)   for(j=0;j<ca;++j)  B[i][j]=A[j][i];   * 1. for(i=0;i<ca;++i)   for(j=0;j<fa;++j)  B[j][i]=A[i][j];   * 1. for(i=0;i<fa;++i)   for(j=0;j<ca;++j)  B[j][i]=A[i][j];   * 1. Ninguna de las anteriores | #include <stdio.h>  #define FIL 10  #define COL 10  typedef double array2d[FIL][COL];  void traspuesta(array2d A, int fa, int ca, array2d B, int \*fb, int \*cb){  int i,j;  \*cb=fa;  \*fb=ca;  /\* CODIGO A INSERTAR \*/  }  int main(){  array2d A={{1,2,3},{4,5,6}};  array2d B;  int fa=2, ca=3,fb,cb;  traspuesta(A,fa,ca,B,&fb,&cb);  … |

* Dada la función **posicion** en C, indicar qué resultado se escribe en pantalla al ejecutar el siguiente programa:

|  |  |
| --- | --- |
| 1. -1 2. 0 3. 1 4. 2 5. 3 6. 4 7. 5 8. 6 9. 7 10. 8 11. 9 12. 10 13. Error de compilación 14. Ninguna de las anteriores | #include <stdio.h>  int posicion(char \*cad, char c){  int pos=-1;  int i=0;  while(cad[i]!='\0'){  if(cad[i]==c){  pos=i;  }  i++;  }  return(pos);  }  int main(){  printf("%d",posicion("EJEMPLO DE", 'E'));  return 0;  } |

* ¿Qué diferencias existen entre los dos constructores de estructuras de datos **struct** y **union** del lenguaje C?
  1. No existe ninguna diferencia, ambos se utilizan para construir registros.
  2. **struct** se utiliza para construir registros planos (sin parte variante), mientras que **union** se utiliza para construir registros con parte variante.
  3. **struct** se utiliza para construir registros que tienen todos sus componentes del mismo tipo de datos, mientras que en el caso de **union** los componentes pueden ser de tipos de datos diferentes.
  4. Los componentes de las estructuras de datos que se construyen con **struct** se representan en MC en posiciones independientes del segmento de pila (“stack”), mientras que en el caso de **union** se representan en el segmento de “heap”, reservando el compilador memoria suficiente para todos ellos.
  5. Los componentes de las estructuras de datos que se construyen con **struct** se representan en MC en posiciones independientes, mientras que en el caso de **union** todos se representan en las mismas posiciones, reservando el compilador memoria suficiente para el mayor de todos ellos.
  6. Ninguna de las respuestas anteriores es correcta.
* De las siguientes declaraciones de nuevos tipos de datos, indicar cuál o cuáles se podrían seleccionar para representar una lista de 10 puntos tridimensionales como máximo:
  1. typedef float tipoPuntos[10][10][10];

typedef struct{

int n;

tipoPuntos p;

}tipoLista;

* 1. typedef struct{

int n;

float x[10],y[10],z[10];

}tipoLista;

* 1. typedef struct{

float x,y,z;

}tipoPunto;

typedef struct{

int n;

tipoPunto p[10];

}tipoLista;

* 1. typedef struct{

float x,y,z;

tipoPunto;

typedef tipoPunto tipoVector[10];

typedef struct{

int n;

tipoVector p;

}tipoLista;

* 1. Ninguna de las anteriores
* Dado el siguiente fragmento de código en lenguaje C indique cuál o cuáles de los siguientes bloques de instrucciones son correctos (tanto sintáctica como semánticamente):

|  |  |
| --- | --- |
| * 1. v1.p.x=1.5;   2. v1.p.x[0]=1.5;   3. v1[0].p.x=1.5;   4. v1.p[0].x=1.5;   5. v2=v1;   6. \*v3=v2;   7. v2[0].p[1].x=2.2;   8. \*v3=v1;   v3->p[0]->x=3.9;   * 1. \*v3=v1;   v3.p[0].x=3.9;   * 1. \*v3=v2[0];   2. Ninguno de los anteriores | #include <stdio.h>  #include <math.h>  typedef struct{  float x,y,z;  }tipoPunto;  typedef struct{  int n;  tipoPunto p[100];  }tipoLista;  int main(void){  tipoLista v1,v2[10],\*v3;  v1.n=3;  v3=&v1;  ...  return 0;  } |

* ¿Cuántas iteraciones requiere el bucle del algoritmo de búsqueda binaria para localizar la posición de inserción (primer elemento de clave estrictamente menor) de un nuevo elemento de clave=1 en el vector V?

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| V | 9 | 9 | 6 | 7 | 7 | 7 | 6 | 6 | 5 | 5 | 4 | 3 | 1 | 1 | 1 | 1 | 0 |

* 1. 3
  2. 4
  3. 5
  4. 6
  5. 7
  6. No se puede aplicar el algoritmo de búsqueda binaria a este vector
* En lenguaje de programación C, la instrucción ***fopen("datos.txt","a");***, cuál o cuáles de las siguientes acciones realiza:

1. Abre el archivo "datos.txt" situado en el directorio actual en modo lectura y devuelve un puntero FILE no nulo si la operación tiene éxito o bien un puntero NULL si la operación no tiene éxito (probablemente porque no existe el archivo).
2. Abre el archivo "datos.txt" situado en el directorio actual en modo escritura y devuelve un puntero FILE no nulo si la operación tiene éxito o bien un puntero NULL si la operación no tiene éxito (probablemente porque no existe el archivo).
3. Crea el archivo "datos.txt" situado en el directorio actual y lo abre seguidamente en modo escritura, devolviendo un puntero FILE no nulo. En caso de existir el archivo, se destruyen sus contenidos (sin avisar) y se crea vacío de contenidos.
4. Abre el archivo "datos.txt" situado en el directorio actual en modo escritura y devuelve un puntero FILE no nulo si la operación tiene éxito. En el caso de no existir el archivo en el directorio actual, se crea previamente.
5. No existe la función **fopen** en ninguna de las librerías estándar de C.
6. Ninguna de las respuestas anteriores es correcta.

* El archivo **“datos.txt”** situado en el directorio actual contiene las temperaturas máximas diarias registradas en una estación meteorológica a lo largo de un mes. Cuál o cuáles de los siguientes fragmentos de programa utilizaría para calcular correctamente la temperatura media de las máximas registrada en ese periodo:

datos.txt={23.5 25.1 26.7 24.2 25.9 30.1 29.0 29.2 29.4 31.6 30.8 28.8 27.5 24.4 23.2 21.9 25.7 28.1 30.1 31.3 }

1. fscanf(f," %f",x);

#include <stdio.h>

int main(){

FILE \*f;

float x, suma=0;

int n=0;

if((f=fopen("datos.txt","r"))==NULL){

printf("Error al abrir archivo");

}else{ /\* FRAGMENTO DE PROGRAMA \*/

fclose(f);

if(n){ printf("Media: %.1f\n",suma/n);

}else{ printf("Sin datos\n");

}

}

return 0;

}

while(!feof(f)){

suma+=x;

n++;

fscanf(f," %f",x);

}

1. while(!feof(f)){

fscanf(f," %f",&x);

suma+=x;

n++;

}

1. fscanf(" %f",&x);

while(!feof(f)){

suma+=x;

n++;

fscanf(" %f",&x);

}

1. while(!feof(f)){

suma+=x;

n++;

fscanf(" %f",x);

}

1. Ninguna las anteriores

* Tenemos un archivo de texto en el directorio actual llamado “temperatura.txt” conteniendo cada línea un registro de la temperatura medida en cada instante de tiempo con el siguiente formato: Hora (entero) y temperatura (real), ambos datos separados por al menos un espacio en blanco. Considerando las siguientes declaraciones, si queremos leer un registro del mismo, indicar qué instrucción o instrucciones son las correctas:

1. fscanf(f, "%d %f", reg);

#include <stdio.h>

typedef struct{

int h;

float t;

}tipo\_reg;

int main(){

FILE \*f;

tipo\_reg reg;

if((f=fopen("temperatura.txt","r"))!=NULL){

…

1. fscanf(f, "%d %f", &reg);
2. fscanf(f, "%d %f", reg.h,reg.t);
3. fscanf(f, "%d %f", reg->h,reg->t);
4. fscanf(f, "%d %f", &reg.h,&reg.t);
5. fscanf(f, "%d %f", &reg->h,&reg->t);
6. Ninguna de las anteriores es correcta