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

**1 de Julio 2021**

**APELLIDO/S: NOMBRE:**

**DNI: GRUPO DOCENTE: GRUPO TRABAJO:**

**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)

* ¿Cuál ó cuáles de las siguientes afirmaciones son correctas?

1. La representación en C de los datos **long int** siempre ocupa el doble de almacenamiento de los datos **int**
2. La representación en C de un número real con el tipo **double** es exacta y sin límites
3. La representación en C de un dato con el tipo **char** siempre ocupa 1 byte de almacenamiento
4. La representación en C de los datos lógicos se hace a través de otros tipos de datos (**int** o **char**)
5. Todas las afirmaciones anteriores son incorrectas

* ¿Cuál o cuáles de las siguientes constantes literales son válidas en lenguaje C?

1. 0
2. 1.E-3
3. ' \101'
4. "Stop: '1' peligro"
5. 0xA73F
6. "0xA73F"
7. -1.0E4.2
8. true
9. '/n'
10. "/n"
11. Ninguna de las anteriores es válida

* Dado el siguiente código en lenguaje C, señale la afirmación o afirmaciones correctas:

|  |  |
| --- | --- |
| * 1. El programa lee por teclado el número del DNI y luego una letra, tras lo cual imprime el NIF   2. Error de compilación: falta la librería stdlib.h   3. Error de compilación: falta un ; tras llave de cierre }   4. El programa tiene un error en tiempo de ejecución: no se puede restar un entero a un carácter (%i-%c)   5. Todas las afirmaciones anteriores son incorrectas | #include <stdio.h>  int main(void){  int num;  char c;  printf("Introduzca numero DNI: ");  scanf(" %i", num);  printf("Introduzca letra: ");  scanf(" %c",c);  printf("NIF: %i-%c", num, c);  return 0;  } |

* Indique cuál (o cuáles) de las siguientes afirmaciones es verdadera en lenguaje C:
  1. Una variable de tipo **unsigned int** puede almacenar valores negativos.
  2. Una variable de tipo **double** no puede almacenar valores enteros.
  3. La sentencia “char letras='ab';” asigna a la variable **letras** la cadena de caracteres 'ab'.
  4. La sentencia “double if=3.2;” asigna a la variable **if** el valor **3.2**.
  5. Todas las afirmaciones anteriores son falsas
* En lenguaje C, ¿qué instrucción/instrucciones de las siguientes utilizaría para escribir en pantalla un número real en doble precisión con 3 cifras decimales almacenado en la variable **x**?
  1. printf("%3f", x);
  2. printf("%.3f", x);
  3. printf("%f3", x);
  4. printf("%f3.3", x);
  5. printf("%3f", \*x);
  6. Ninguna de las anteriores
* En C, si tenemos que **a**, **b** y **c** son variables de tipo **int**, **float** y **char**, respectivamente, y que tienen asignados los valores a=10, b=5.1 y c='x', indicar qué expresión (o expresiones) devuelven como resultado verdadero (distinto de cero):
  1. !((a<b)&&(c=='x'))
  2. (a=(a=20)%5) || ! (a && ! c)
  3. a && ! c || b
  4. (b-->5.0)&&(++a>10)
  5. Ninguna de las anteriores
* ¿Qué mensaje aparece en pantalla al ejecutar el siguiente programa en C?

|  |  |
| --- | --- |
| * 1. Agua hirviendo   2. Agua muy caliente   3. Agua caliente   4. Agua congelada   5. (No aparece nada en pantalla)   6. Error de compilación: se asigna un valor entero a una variable real   7. Error de ejecución: el programa entra en un bucle infinito | #include <stdio.h>  int main(){  double temp=15;  if(temp>0)  if(temp>=100)  printf("Agua hirviendo");  else if(temp>50)  printf("Agua muy caliente");  else if(temp>25)  printf("Agua caliente");  else printf("Agua congelada");  return 0;  } |

* Indicar los valores que imprime en pantalla el siguiente programa en C:

|  |  |
| --- | --- |
| * 1. 10 9 8 7 6 5 4 3 2 1 0   2. 10 9 8 7 6 5 4 3 2 1   3. 9 8 7 6 5 4 3 2 1 0 -1   4. 9 8 7 6 5 4 3 2 1   5. Error de compilación | #include <stdio.h>  int main(){  int a=10, b;  while(a>0){  b=a--;  printf("%i ", b);  }  return 0;  } |

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

|  |  |
| --- | --- |
| * 1. 16 64   2. 20 144   3. 80 4096   4. 4096 16777216   5. Error de compilación   6. Ninguna de las respuestas anteriores es correcta | #include <stdio.h>  int main(){  int a=1,b=1,mayor;  while((a<40)&&(b<100)){  if(a>b) mayor=a;  else mayor=b;  a=a+mayor;  b=b\*mayor;  }  printf("%i %i",a,b);  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 | #include <stdio.h>  int main(void) {  int 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;  } |

* ¿Cuál (o cuáles) de las siguientes afirmaciones es cierta en relación con el prototipo de una función en C?

1. Es obligatoria su inclusión al principio del código fuente
2. Es obligatoria su inclusión al final del código fuente
3. Puede ser eliminado sin ninguna consecuencia en el orden de escritura de las definiciones de las funciones en el código fuente
4. En lenguaje C está prohibida su utilización a menos que se incluya la librería **stdlib.h**
5. Ninguna de las respuestas anteriores es cierta

* En lenguaje C, ¿cuál o cuáles de las siguientes afirmaciones es correcta cuando se considera el siguiente prototipo de función? void Uno(Dos Tres, Cuatro \*Cinco);
  1. **Uno** es un identificador de función
  2. **Tres** y **Cinco** pueden ser el mismo identificador
  3. **Uno** es un identificador de variable
  4. **Dos** y **Cuatro** pueden ser el mismo identificador
  5. Error de sintaxis: la función debe devolver un tipo de dato distinto de **void**
* ¿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. 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=%f y=%f",x,y);  return 0;  }  void intercambiar(double \*x, double \*y){  \*x=\*y;  \*y=\*x;  } |

* 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); 4. leer\_entero\_rango(0,\*n,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 \*/  } |

* Al ejecutar el siguiente programa en C, ¿qué resultado aparecerá en pantalla?

|  |  |
| --- | --- |
| 1. 4 2. 6 3. 8 4. Error de sintaxis en función suma 5. Error en tiempo de ejecución: bucle infinito | #include <stdio.h>  int suma(int n);  int main(){  printf("%d",suma(suma(2)));  return(0);  }  int suma(int n){  int i=0,s=0;  while(i<=n){  s+=i;  ++i;  }  return(s);  } |

* 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. Error de compilación 5. Error de ejecución: bucle infinito | int leer\_entero\_intervalo(int a, int b){  int x;  do{ printf("Numero [%d,%d]: ",a,b);  scanf(" %i",&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. f(x,y,c,\*z); 2. z=f(x,y,c); 3. f(x+y,3,c,z); 4. f(y,x,'A',z); 5. Todas las anteriores son incorrectas. | #include <stdio.h>  void f(int x, int y, char c, float \*z);  int main(){  int x=4, y=5;  char c='C';  float \*z;  -------------------  return(0);  }  void f(int x, int y, char c, float \*z){  /\* Definición de función \*/  } |

* Considerar el siguiente programa en C, donde la función llamada hace uso de la variable global (o externa) **a**. Después de la ejecución de este programa:

|  |  |
| --- | --- |
| 1. Variable a = 0 2. Variable a = 1 3. Variable a = 2 4. Error de compilación: el lenguaje C prohíbe el uso de variables externas. 5. Todas las respuestas anteriores son incorrectas | #include <stdio.h>  int a=1;  void proc (int \*b, int \*c){  \*b=3;  \*c=a%2;  \*b=a+(\*b)\*(\*c);  }  int main(){  proc(&a,&a);  printf("Variable a = %d", a);  return 0;  } |

* Indicar qué instrucciones hay que sustituir en lugar del comentario para que el siguiente programa imprima la representación en binario del número entero **n** que se recibe como argumento:

|  |  |
| --- | --- |
| 1. printf(" %d" , n%2);   imprime\_binario (n/2);   1. imprime\_binario (n/2);   printf(" %d" , n%2);   1. imprime\_binario (n/2);   printf(" %d" , n/2);   1. imprime\_binario (n%2);   printf(" %d" , n%2); | void imprime\_binario(int n){  if(n>=2){  /\* COMENTARIO A SUSTITUIR \*/  }else printf(" %d" , n);  } |

* Considere el siguiente prototipo de una función C que calcula la longitud de un tramo de una curva alabeada paramétrica cualquiera comprendido entre los valores del parámetro ***t=t1*** y ***t=t2***:

double longCurva(void (\*c)(double t,double \*x,double \*y,double \*z),

double t1, double t2);

Si el siguiente prototipo de función se corresponde con la implementación en C de la hélice cilíndrica ***c(t)=(10\*cos(t),-10\*seno(t),0.5\*t)***:

void c(double t,double \*x,double \*y,double \*z);

¿Cuál o cuáles de las siguientes instrucciones presenta en pantalla la longitud del tramo de dicha curva comprendido entre ***t=0*** y ***t=20\*π***?

1. printf(" Longitud= %f" ,longCurva(c(t,&x,&y,&z),0,20\*M\_PI));
2. printf(" Longitud= %f" ,longCurva(c(t,\*x,\*y,\*z),0,20\*M\_PI));
3. printf(" Longitud= %f" ,longCurva(\*c,0,20\*M\_PI));
4. printf(" Longitud= %f" ,longCurva(c,0,20\*M\_PI));
5. Ninguna de las anteriores

* ¿Cuál (o cuáles) de las siguientes instrucciones lee un carácter desde teclado en la posición i-ésima de un “array” unidimensional de caracteres en C de nombre **v**?
  1. scanf(" %c", v[i]);
  2. scanf(" %c", &v[i]);
  3. scanf(" %c", \*v[i]);
  4. v[i] = getchar();
  5. Ninguna de las anteriores
* De las siguientes declaraciones de nuevos tipos de datos, indicar cuál o cuáles seleccionaría para representar el número total de vueltas que da una rueda en un intervalo de tiempo de 100 segundos. La rueda está inicialmente en reposo y se somete a una aceleración angular constante **α** durante 10 segundos: tipo a;

1. typedef double tipo[1..100];
2. typedef double tipo[99];
3. typedef double tipo[100];
4. typedef double tipo[101];
5. Ninguna de las anteriores

* Dadas las variables A y B de tipo “array” unidimensional de diez reales en doble precisión, ¿cuál o cuáles de las siguientes instrucciones utilizaría para copiar todos los elementos de A en B en lenguaje C?
  1. A=B;
  2. B=A;
  3. for(int i=0; i<10;++i){

B[ i ] = A[ i ];

}

* 1. for(int i=1; i<=10; i++){

B[ i ] = A[ i ];

}

* 1. for(int i=0; i<10; ++i){

A[ i ] = B[ i ];

}

* De las siguientes declaraciones de nuevos tipos de datos, indicar cuáles seleccionaría para representar las posiciones espaciales en coordenadas cartesianas de un avión en 100 instantes de tiempo diferentes: tipo a;
  1. typedef double tipo[0..99,0..99,0..99];
  2. typedef double tipo[99][99][99];
  3. typedef double tipo[100][100][100];
  4. typedef double tipo[101][101][101];
  5. Ninguna de las anteriores
* ¿Cuál o cuáles de los siguientes fragmentos de código en C utilizaría para sumar todos los elementos de un “array” bidimensional de reales en doble precisión del tipo de datos indicado?

|  |  |
| --- | --- |
| * 1. for(i=0;i<FIL;++i)   for(k=0;k<COL;++k)  suma+=m[i][k];   * 1. for(i=0;i<COL;++i)   for(k=0;k<FIL;++k)  suma+=m[i][k];   * 1. for(i=1;i<=COL;i++)   for(k=1;k<=FIL;k++)  suma+=m[i][k];   * 1. for(i=0;i<COL;++i)   for(k=0;k<FIL;++k)  suma+=m[k][i];   * 1. Ninguna de las anteriores | #include <stdio.h>  #define FIL 10  #define COL 20  typedef double tipo\_matriz[FIL][COL];  …  tipo\_matriz m;  int i,k;  double suma=0;  … |

* Indicar cuál o cuáles de las siguientes funciones presenta por pantalla la cadena de caracteres que se pasa como argumento, pero escrita al revés:
  1. void escribir (char \*n){

if (\*n!=0) {

escribir (n+1);

putchar(\*n);

}

}

* 1. void escribir (char \*n){

if (\*n!=0){

escribir (n++);

putchar (\*n);

}

}

* 1. void escribir (char \*n){

if (\*n!='\0'){

escribir (n+1);

putchar (\*n);

}

}

* 1. void escribir (char \*n){

if (\*n!=0){

escribir (++n);

putchar (n);

}

}

* 1. Ninguna de las anteriores
* Dadas las siguientes declaraciones en lenguaje C, indique cuál o cuáles de las siguientes instrucciones es correcta:

|  |  |
| --- | --- |
| * 1. printf("%f %f", \*p.real, \*p.imag);   2. scanf(" %f %f", p->real, p->imag);   3. scanf(" %f %f", &p->real, &p->imag);   4. scanf(" %f %f", x[0]->real, x[0]->imag);   5. Todas las instrucciones anteriores son incorrectas | typedef struct{  float real;  float imag;  }complejo;  complejo x[10], \*p;  p=&x[0]; |

* 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. p2=p1;   2. p3=p1;   3. \*p3=p1;   4. p3=&p1;   \*p3=p2;   * 1. p1->x=75.0;   p1->y=45.3;   * 1. p1.tipo=1;   p1.r=sqrt(p1.x\*p1.x+p2.x\*p2.x);  p1.z=atan2(p1.y/p1.x);   * 1. p3=&p1;   p3.tipo=1;  p3.x=50.0;  p3.y=M\_PI/2;   * 1. Ninguno de los anteriores | #include <stdio.h>  #include <math.h>  typedef struct{  int tipo; /\* 0: cartesianas, 1: polares \*/  union{  struct{  double x,y;  };  struct{  double r,z;  };  };  }tipo\_coordenadas;  int main(void){  tipo\_coordenadas p1,p2,\*p3;  p1.tipo=0;  p1.x=100.0;  p1.y=100.0;  ...  return 0;  } |

* De las siguientes declaraciones de nuevos tipos de datos, indicar cuál o cuáles seleccionaría para representar una imagen digitalizada de 1024\*1024 cuadrículas, que contiene el color de cada cuadrícula en formato RGB (rojo, verde y azul) mediante 3 números reales en el intervalo [0,1]:

|  |  |
| --- | --- |
| * 1. typedef tipoColor tipoImagen[NF][NC];   typedef struct{  double r,g,b;  }tipoColor;   * 1. typedef struct{   double r,g,b;  }tipoColor;  typedef tipoColor tipoImagen[NF][NC];   * 1. typedef struct{   double r,g,b;  }tipoColor;  typedef tipoImagen tipoColor[NF][NC];   * 1. Ninguna de las anteriores | #define NF 1024  #define NC 1024  /\* Definición de tipoImagen \*/  tipoImagen a; |

* ¿Cuántas iteraciones requiere el bucle del algoritmo de búsqueda binaria para localizar la posición de inserción (primer elemento de clave estrictamente mayor) de un nuevo elemento de clave=15 en el vector V?

0 1 2 3 4 5 6 7 8 9 10 11 12

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| V | 1 | 8 | 10 | 11 | 15 | 15 | 15 | 15 | 17 | 29 | 31 | 31 | 40 |

* 1. 3
  2. 4
  3. 5
  4. 6
  5. 7
  6. No se puede aplicar el algoritmo de búsqueda binaria a este vector