Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) ¿Qué sintaxis empleamos en Maxima para definir una lista?
2. (1 punto) ¿Qué comando se emplea en Maxima para cargar un "paquete de funciones"? Como p. ej, el paquete "dynamics" que incluye (entre otras) la función rk.
3. (2 punto) El comando `rk` (Runge-Kutta) del Maxima, ¿sirve para resolver ecuaciones diferenciales ordinarias de orden superior a 1? Explique brevemente cómo.

Programación en C

1. (2 puntos) Indique qué guarda la siguiente función en el archivo.

```
void comoCSV(char* fn, char** cols, int n, int m, double** vals) {
    FILE *f=fopen(fn, "wt");
    int i, j;
    fprintf(f, "%s", cols[0]);
    for(j=0; j<m; j++) {
        fprintf(f, ",%s", cols[j]);
    }
    fprintf(f, "\n");
    for(i=0; i<n; i++) {
        fprintf(f, "%g", vals[i][0]);
        for(j=1; j<m; j++) {
            fprintf(f, ", %g", vals[i][j]);
        }
        fprintf(f, "\n");
    }
    fclose(f);
}
```

2. (4 puntos) El método de Newton-Raphson es un método numérico para resolver ecuaciones algebraicas no-lineales. Este método está basado en el hecho de que las soluciones de la ecuación $h(x) = g(x)$ son las raíces de la función $f(x) = h(x) - g(x)$. Para obtener numéricamente estas raíces se utiliza el algoritmo iterativo de Newton-Raphson:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

donde $f'(x_n)$ es la derivada de la función evaluada en el punto x_n.

Partiendo de una semilla inicial x_0 la sucesión convergerá (no siempre, dependerá de las características de la ecuación y de la semilla de partida) a una de las raíces de la función $f(x)$. En general el algoritmo concluye cuando el error relativo entre dos aproximaciones sucesivas es menor que una cierta tolerancia fijada, que llamaremos E, que se considera como el error relativo de la aproximación:

$$\frac{|x_{n+1} - x_n|}{|x_{n+1}|} < E$$

El objetivo de este ejercicio es escribir un programa que calcule las soluciones reales de la ecuación $e^x = x^3$. La función correspondiente cuyas raíces queremos obtener, y su derivada deberán ser definidas en dos funciones. Los valores de la semilla y del error serán introducidos por línea de comandos (como argumentos de la función "main") cuando se ejecute el programa. El programa deberá imprimir en pantalla el valor aproximado de la raíz obtenida con esa tolerancia.

2017-S1.pdf

Thegoodking

Física Computacional I

1º Grado en Física

Facultad de Ciencias
Universidad Nacional de Educación a Distancia

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) Explique qué se obtiene y qué sale por pantalla después de evaluar este código:

   ```
   a : tan(pi)$
   ```

   Respuesta: Si la variable pi no ha sido definida (recuérdese que el número π se representa en máxima por %pi), el lado derecho del signo de asignación (:) se dejará sin evaluar, como tan(pi) y esta expresión será asignada a la variable a. Como la instrucción finaliza con **$**, no se mostrará salida por consola.

2. (1 punto) Después de evaluar este código

   ```
   f(x):=block([a],a:3,x^a);
   ```

   ¿cuál será el valor de la variable a?

   Respuesta: Este código define una función, por lo tanto, en tanto no se evalúe, no afectará al valor de ninguna variable. De todos modos, aún si se ejecutase como

   ```
   f(x);
   ```

   la variable a sería una variable local (se incluye en el corchete que va como primer argumento de block()), con valor sólo dentro de la función, por lo que al ejecutar ésta, a seguirá valiendo lo que valía antes de ejecutarla.

3. (2 puntos) La identidad $e^{i\pi} + 1 = 0$, debida a Euler, contiene todos los números importantes de las matemáticas. Escriba una instrucción condicional en Maxima que diga ‘‘`Euler tenía razón`’’, en caso de que la igualdad anterior sea cierta, o ‘‘`Euler estaba equivocado`’’, en caso de que, para Maxima, no sea así. **Nota**: para emitir los mensajes, utilice la función `disp(‘‘cadena’’)`.

   Respuesta: En Maxima las función condicional es de la forma

   ```
   if condicion then consecuencia else alternativa
   ```

   así que escribiríamos

   ```
   if exp(%i*%pi)+1=0 then disp("Euler tenia razon") else disp("Euler estaba
       equivocado")$
   ```

Programación en C

1. (6 puntos) El objetivo de este ejercicio es diseñar un programa que calcule el centro de masas de un sistema discreto de partículas.

   - El programa debe estar compuesto de una única función "main".
   - Supondremos que los datos de las N partículas se encuentran en un archivo con cuatro columnas: las tres primeras columnas corresponden a las coordenadas (x_i, y_i, z_i) de cada partícula i, mientras que la cuarta columna corresponde a la masa m_i. Aunque no conocemos exactamente el número de partículas que componen el sistema, sabemos que es menor que 1000. El nombre del archivo que contiene los datos deberá ser introducido por línea de comandos (como argumento de la función "main") cuando se ejecute el programa.
   - Para leer y almacenar los datos de las partículas, se deberá definir una estructura denominada "Punto" que representa una partícula con su posición y masa. Después, en la función "main" de deberá declarar un "array" de "Puntos", es decir, un array de estas estructuras, para almacenar en él la información de todo el sistema de partículas.
   - El programa debe escribir en pantalla la posición del centro de masas del sistema

$$\boldsymbol{r}_{CM} = \frac{\sum_{i=1}^{N} m_i \boldsymbol{r}_i}{\sum_{i=1}^{N} m_i}$$

Solución:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define N 1000

struct punto {
        double x,y,z,masa;
};

typedef struct punto Punto;

int main(int argc, char** argv) {
    int i, n, npts;
    double x, y, z, masa, masaCM;
    Punto P[N];
    Punto PCM;
    FILE *fin;

    fin=fopen(argv[1], "r");
    npts=0;

    do {
        n=fscanf(fin,"%f %f %f %f", &x, &y, &z, &masa);
        if (n==4) {
            P[npts].x=x;
            P[npts].y=y;
            P[npts].z=z;
            P[npts].masa=masa;
            npts++;
```

```
        }
    } while(n==4);

    fclose(fin);

    PCM.x=0;
    PCM.y=0;
    PCM.z=0;
    masaCM=0;

    for(i=0; i<npts; i++) {
        PCM.x+=(P[i].masa)*(P[i].x);
        PCM.y+=(P[i].masa)*(P[i].y);
        PCM.z+=(P[i].masa)*(P[i].z);
        masaCM+=P[i].masa;
    }

    printf("La posicion del Centro de Masas es (%g, %g, %g)\n",
        PCM.x/masaCM, PCM.y/masaCM, PCM.z/masaCM);

    return 0;
}
```

2017-S2.pdf

Thegoodking

Física Computacional I

1º Grado en Física

Facultad de Ciencias
Universidad Nacional de Educación a Distancia

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) ¿Qué caracteres empleamos en Maxima para definir una lista?

   Respuesta: Se emplean "[", "]" y ",". Los dos primeros indican el comienzo y final de lista, respectivamente, mientras que la coma se emplea para separar los elementos de la lista.

2. (1 punto) Explique qué se obtiene y qué sale por pantalla después de evaluar este código:

```
a : tan(%pi)$
b : a * c^2;
```

   Respuesta: La primera instrucción asigna a la variable a el valor 0, ya que éste es el valor de la tangente de π; esto es, evalúa el lado derecho del operador de asignación y asigna ese valor a la variable que figura al lado izquierdo. Como la instrucción está terminada por un **$**, no produce ninguna salida en la consola.

   La segunda instrucción asigna a la variable b el valor resultante de evaluar a$*$c^2. Dado que en la instrucción anterior se asignó a la variable a el valor 0, el resultado será 0. Ahora, como la instrucción finaliza con un ;, se mostrará el valor en la consola.

3. (2 puntos) Escriba una función en Maxima que reciba como argumentos una lista de coeficientes (como p.ej. $[a_0, a_1, a_2, \ldots a_n]$) y una variable (como p.ej. x) y que construya a partir de este *input* el polinomio en la variable dada

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

   Respuesta: Llamemos a la función Pa(as) donde as es la lista de coeficientes; el número de coeficientes (que es el grado del polinomio más uno, vale length(as)). La forma más sencilla de calcular el polinomio es mediante la función sum() aplicada a los monomios $a_i x^i$:

```
Pa(as):=sum( as[i]*x^i, i, 0, length(as)-1 );
```

Programación en C

1. (2 puntos) Escribir una función que, dada una lista de números complejos en la forma de un array de estructuras

```
struct complexf {
        float re, im;
};
```

devuelva el número complejo resultante de multiplicar todos ellos.

Solución:
```
struct complexf complex_prod(int N, struct complexf x[]) {
        int i;
        float aux;
        struct complexf p;
                p.re=1.0;
                p.im=0.0;
        for(i=0; i < N; i++) {
                aux=p.re;
                p.re=p.re*x[i].re-p.im*x[i].im;
                p.im=aux*x[i].im+p.im*x[i].re;
        }
        return p;
}
```

2. (4 puntos) Supongamos que tenemos un archivo de datos "datos.dat" con N pares de puntos que representan el comportamiento de una función $f(x)$ en un intervalo dado. La primera columna del archivo corresponde a los valores x_i (que están ordenados de menor a mayor) mientras que la segunda muestra los valores de la función $f(x_i)$.

   Escribir un programa que lea esos puntos y obtenga los valores de x en los que la función muestra los mínimos y máximos relativos dentro del intervalo. El valor de N debe ser introducido por línea de comandos como argumento de la función "main" cuando se ejecuta el programa. El programa debe mostrar en pantalla el resultado del siguiente modo (es un ejemplo):

   ''La función tiene tres mínimos relativos en los puntos x = 0.1, 1.2, 4.5''
   ''La función tiene dos máximos relativos en los puntos x = 0.9, 2.9''

   Del mismo modo el programa también deberá indicar si la función no tiene mínimos ni máximos relativos.

Solución:
```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv) {
    int N, i, nmin, nmax;
    N=atoi(argv[1]);
    FILE *fin;
    double x[N],y[N], xmin[N], xmax[N];

    fin=fopen("datos.dat", "r");
    for (i=0; i<N; i++) {
        fscanf(fin, "%f %f", &x[i], &y[i]);
    }
    fclose(fin);

    nmin=nmax=0;

    for(i=1; i<N-1; i++) {
        if(y[i-1]>y[i] && y[i+1]>y[i]) {
            xmin[nmin]=x[i];
            nmin++;
```

```
        }
        if(y[i-1]<y[i] && y[i+1]<y[i]) {
            xmax[nmax]=x[i];
            nmax++;
        }
    }

    if(nmin==0)
        printf("La funcion no tiene ningun minimo relativo\n");
    else {
        printf("La funcion tiene %d minimos relativos"
            "en los puntos x = ", nmin);
        for (i=0; i<nmin; i++)
            printf("%g, ", xmin[i]);
        printf("%\n");
    }

    if (nmax==0)
        printf("La funcion no tiene ningun maximo relativo\n");
    else {
        printf("la funcion tiene %d maximos relativos"
            "en los puntos x = ", nmax);
        for (i=0; i<nmax; i++)
            printf("%g, ", xmax[i]);
    }

    return 0;
}
```

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) Abrimos una sesión de trabajo en Maxima y tecleamos

   ```
   a:i^2$
   b:1+a;
   ```

   Indique el output que se obtiene por pantalla.

2. (1 punto) Abrimos una sesión de trabajo en Maxima y tecleamos

   ```
   a:trigsimp(cos(x)^2 + sin(x)^2)$
   b:sqrt(-a);
   ```

   Indique el output que se obtiene por pantalla.

3. (2 puntos) El *número áureo* (también llamado *proporción áurea*) se define como la mayor de las raices de $x+1=x^2$, dada por

   $$\phi = \frac{1+\sqrt{5}}{2} \simeq 1{,}62$$

   Este número representa una relación de tamaños estéticamente afortunada. Dado un rectángulo cuyos lados tienen longitudes a y b, escriba una función en Maxima que reciba dichas longitudes como argumentos, calcule la razón de aspecto del rectángulo (el mayor de los valores a y b, dividido por el menor) y devuelva el cociente entre esa razón de aspecto y el valor del número aúreo almacenado en una variable local, `phi`, de la función.

Programación en C

1. (3 puntos) Escriba la implementación de la función

   ```
   double lever(double c0, double c1, double c2, double *x2);
   ```

   que devuelva los valores x_1 (como valor retornado) y x_2 (en el último argumento) tales que se cumpla el sistema de ecuaciones:

   $$x_1 + x_2 = 1$$
   $$c_1 x_1 + c_2 x_2 = c_0$$

2. (3 puntos) La aproximación numérica de 5 puntos para la derivada primera de una función $f(x)$ tiene la forma

$$f'(x) \approx \frac{1}{12h}\left[f(x-2h) - 8f(x-h) + 8f(x+h) - f(x+2h)\right]$$

donde h es una intervalo real pequeño ($h \ll 1$). El error de esta aproximación es, aproximadamente, h^4.

Escriba una función en C que calcule la derivada de una función $f(x)$ en un cierto punto x_0 con una cierta tolerancia que llamaremos `err`. Consideraremos que la función matemática que queremos derivar, $f(x)$, se ha definido previamente en el programa como por ejemplo:

```
double funcion(double x)
{
    double y;
        //...
    return y;
}
```

La función que calcula la derivada debe recibir como argumentos la función matemática que queremos derivar, el punto x_0 en el que queremos obtener la derivada y la tolerancia de la aproximación, `err`. El intervalo h que deberemos emplear estará dado por $h = \texttt{err}^{1/4}$.

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) Abrimos una sesión de trabajo en Maxima y tecleamos

```
a:tan(%pi/4)$
b:sqrt(9/a);
```

   Indique el output que se obtiene por pantalla.

2. (1 punto) Abrimos una sesión de trabajo en Maxima y tecleamos

```
a:log(%e^3)$
b:sqrt(a);
```

   Indique el output que se obtiene por pantalla.

3. (2 puntos) Se define la circularidad de una figura plana como el cociente entre su perímetro y el correspondiente a un círculo de igual área. Consideremos un rectángulo con lados de longitudes a y b. Escriba una función en Maxima que, basándose en esta información, calcule en variables locales el perímetro (P) y el área (A) de dicha figura, el radio y el perímetro del círculo de igual área ($R = \sqrt{A/\pi}$ y $P' = 2\pi R$) y que, finalmente, devuelva el cociente entre ambos perímetros (P/P').

Programación en C

1. (3 puntos) Escriba la implementación de las funciones

```
float resSeries(int N, float R[]);
float resParall(int N, float R[]);
```

   que devuelven los valores de resistencia total de configuraciones de N resistencias (con valores individuales dados por los R_n) en serie o en paralelo, respectivamente.

2. (3 puntos) Supongamos una red cuadrada discreta en la que cada nodo de la red (i, j) con $i = 0, 1, 2, \ldots$ y $j = 0, 1, 2, \ldots$ ha sido definido mediante la estructura:

```
struct punto
{
        int i,j;
};
typedef struct punto Punto;
```

Consideremos ahora que nuestro programa principal calcula dos trayectorias sobre la red descritas mediante sendos vectores de `Puntos` que han sido declarados dentro de `main()` como:

```
Punto camino1[long1];
Punto camino2[long2];
```

Escriba una función que reciba de `main()` estas dos trayectorias con sus respectivas longitudes, y que devuelva el número de puntos de la red que tienen en común.

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) Abrimos una sesión de trabajo en Maxima y tecleamos

   ```
   a:%i^2$
   b:1-a;
   ```

   Indique el *output* que se obtiene por pantalla.

2. (1 punto) Indique qué caracteres empleamos en Maxima para definir una variable de tipo *string*, o cadena de caracteres.

3. (2 puntos) Tenemos un rectángulo delimitado por dos lados paralelos con longitud a y otros dos lados paralelos con longitud b. Escriba una función en Maxima que, basándonos en estos valores a y b, calcule el perímetro y el área de dicha figura, y los asigne a las variables locales P y A, respectivamente, y que, finalmente, devuelva el cociente entre el área y el cuadrado del perímetro.

Programación en C

1. (2 puntos) Explique brevemente el funcionamiento de la función

   ```
   float axon(int x[], int N, int n[], float w[]) {
           float y=0;
           int k;
           for(k=0; k < N; k++)
                   y+=w[k]*X[n[k]];
           return sigma(y);
   }
   ```

   ¿Qué valores posibles pueden albergar los elementos del vector `n[]`? ¿Cuál debería ser la declaración de la función `sigma()` para que funcione todo correctamente?

2. (4 puntos) La rotación de cualquier vector del espacio un ángulo α alrededor del eje Z, viene dada por la siguiente matriz de cambio de base:

$$R_Z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

De este modo, las nuevas componentes $\mathbf{r}' = (x', y', z')$ resultantes de la rotación del vector $\mathbf{r} = (x, y, z)$ un ángulo α alrededor del eje Z, serán el resultado de $\mathbf{r}' = R_Z(\alpha)\mathbf{r}$ que, en componentes, tiene la forma:

$$x' = x\cos\alpha - y\sin\alpha$$
$$y' = x\sin\alpha + y\cos\alpha$$
$$z' = z$$

El objetivo de este ejercicio es diseñar un programa, que llamaremos `rotacion.c`, que realice la rotación de un vector cualquiera alrededor del eje Z. Las componentes del vector y el ángulo de giro deberán ser introducidos por línea de comandos (como argumentos de la función `main`) cuando se ejecuta el programa. Por ejemplo, la rotación del vector $\mathbf{r} = (0{,}2, 1{,}4, -3{,}5)$ un ángulo de $30° \simeq 0{,}5236$ rad alrededor del eje Z se ejecutará como:

```
rotacion.exe 0.2 1.4 -3.5 0.5236
```

La función `main` deberá leer estos datos y mandar las componentes del vector y el ángulo de giro a una función `rotarZ` que devolverá el resultado en el mismo vector que se pasa. La función `rotarZ` se declarará como:

```
void rotarZ(double r[3], double angulo);
```

Finalmente la función main deberá imprimir las componentes del vector rotado.

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (2 puntos) La función de escalón unidad inverso se define como EUI(x) = 0 si x es mayor que cero y 1 en caso contrario. ¿Cómo podríamos programar esta función en el lenguaje de Maxima?
2. (2 puntos) Según la segunda ley de Newton, al aplicar una fuerza F durante un instante muy pequeño de tiempo dt sobre una masa puntual producimos una variación de su momento lineal dada por $dP = F\,dt$, que equivale a una variación en la velocidad de la partícula igual a dP/m (suponiendo que la masa de la partícula, m, no varía durante el intervalo dt). Programe una función en Maxima usando block(...) que tenga como *input* los datos anteriores (m, F, dt), que defina la variación de momento dP como una variable local y que devuelva finalmente la variación de velocidad dP/m de la partícula.

Programación en C

1. (2 puntos) El más irracional de los números es la razón aurea, $\varphi \simeq 1{,}6180339887\ldots$ Este número se puede calcular como la fracción continua

$$\varphi = 1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \ldots}}}}$$

   Escríbase una función en C que calcule el valor de esta constante matemática de la forma iterativa mostrada en esta fracción continua, parando cuando dos iteraciones consecutivas proporcionen valores que se diferencien en menos de 10^{-15}.

2. (4 puntos) Escriba un programa que pida al usuario las coordenadas de dos vectores tridimensionales, las guarde en forma de estructuras

   ```
   struct Vec3f {  float x, y, z; }
   ```

   Y llame a una función (que también deberá implementar)

   ```
   double angVec3f(struct Vec3f a, struct Vec3f b);
   ```

   que determine el ángulo en grados que forman y lo devuelva al usuario imprimiéndolo por pantalla.

   Nota: recuerde que el ángulo α formado por dos vectores $\boldsymbol{a}, \boldsymbol{b}$ viene dado por $\alpha = \arccos\left(\frac{\boldsymbol{a}\cdot\boldsymbol{b}}{\sqrt{(\boldsymbol{a}\cdot\boldsymbol{a})}\sqrt{(\boldsymbol{b}\cdot\boldsymbol{b})}}\right)$ y el producto escalar de dos vectores $\boldsymbol{a}\cdot\boldsymbol{b} = a_x b_x + a_y b_y + a_z b_z$.

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (2 puntos) Programe una función MP(r, v) en Maxima que calcule la media ponderada de una serie de resultados de acuerdo al siguiente...

```
/* Input:
 * r: lista de resultados cuya media queremos calcular.
 * v: lista con el número de veces (peso) que hemos obtenido cada uno de
      estos resultados
 */
```

2. (2 puntos) Sabiendo que el logaritmo de la media geométrica es la media de los logaritmos, emplee la función del ejercicio anterior para programar una función MGP(r, v) en Maxima (de la manera más sencilla posible) que calcule la media geométrica ponderada de los valores r anteriores (suponiendo que todos ellos son positivos), siendo el *input* el mismo que en el ejercicio anterior.

Programación en C

1. (3 puntos) Los números primos son aquellos números naturales mayores que 1, cuyos únicos divisores (el resto de la división es cero) son ellos mismos y el 1. Por ejemplo, el 5 es primo ya que sus únicos divisores son el 1 y el 5. El 6 no es primo ya que, aparte del 1 y del 6, tiene como divisores el 2 y el 3. Escriba un programa en C que calcule e imprima en un archivo, los N primeros números primos. Tanto el valor de N como el nombre del archivo de texto en el que imprimirán los números primos, deberán ser introducidos por línea de comandos como argumentos de la función `main()`.

2. (3 puntos) Sabemos que una variable aleatoria η normalmente distribuida (según la distribución $N(0;1)$, de media 0 y desviación típica 1) se puede simular como $\eta = \frac{12}{N}\sum_{i=1}^{N}\left(\mu_i - \frac{1}{2}\right)$, con N un número grande (p.ej. 15 o 30) y μ_i valores calculados de una variable aleatoria uniformemente distribuida en el intervalo $[0,\ 1)$. De la misma forma se puede simular una variable aleatoria χ distribuida según la distribución chi-cuadrado con n grados de libertad como $\chi = \sum_{j=1}^{n}\eta_j^2$, con η_j valores calculados de una variable aleatoria distribuida gaussianamente.

   Asumiendo que ya disponemos de una función **double** uniforme() que genera un número aleatorio distribuido uniformemente en el intervalo $[0,\ 1)$, escriba una función **double** chi2(**int** n) que genere un número aleatorio distribuido según la distribución chi-cuadrado con n grados de libertad.

Lea atentamente estas instrucciones antes de comenzar:

- Responda a cada una de las cuestiones prácticas.
- Es necesario aprobar este examen (5 puntos) para aprobar la asignatura.
- No es suficiente este examen para aprobar la asignatura: se deben aprobar los dos trabajos de la parte práctica.

Sistemas de álgebra computacional: Maxima

1. (1 punto) Explique cuál es el resultado obtenido al ejecutar en Maxima esta instrucción:

```
a : tan(pi/4)$
b : sqrt(-a);
```

2. (1 punto) Explique cuál es el resultado obtenido al ejecutar en Maxima esta instrucción:

```
a : exp(%i * %pi)$
b : (1 + a) * (1 - a);
```

3. (2 puntos) La función de escalón unidad se define como EU(x) = 1 si x es mayor o igual que cero y cero en caso contrario. ¿Cómo podríamos programar esta función en el lenguaje de Maxima?

Programación en C

1. (4 puntos) El *teorema de Bolzano* del análisis matemático establece que si una función continua $f(x)$ toma valores de distinto signo en los extremos de un intervalo $[a, b]$, es decir $f(a) \cdot f(b) < 0$, entonces esa función tiene al menos una raiz en dicho intervalo, es decir, existe al menos un valor $x_0 \in [a, b]$, tal que $f(x_0) = 0$. Este teorema es la base teórica del *método de bisección*, un método numérico iterativo para encontrar las raíces de una función. Supongamos que tenemos el intervalo anterior en el que se cumplen las premisas del teorema de Bolzano. En la primera iteración se calcula el punto medio del intervalo x_m y se evalúa la función en ese punto. Si $f(x_m) = 0$ entonces hemos encontrado la raíz buscada. Si no es así, debemos determinar en cuál de los dos subintervalos la función cambia de signo. De este modo, si $f(a) \cdot f(x_m) < 0$ nos quedamos con el intervalo $[a, x_m]$, y si $f(b) \cdot f(x_m) < 0$ nos quedamos con el intervalo $[x_m, b]$. Si repetimos el proceso de forma iterativa sobre el intervalo resultante iremos acotando el punto donde la función se anula. Asumiremos que la función sólo tiene una raíz en el intervalo inicial, de modo que sólo una de las dos condiciones se cumplirá en cada paso. Después de n iteraciones, nuestro intervalo tendrá la forma $[a_n, b_n]$, con $a_0 = a$ y $b_0 = b$, y nuestra aproximación numérica de la raiz vendrá dada por el punto medio $x_0 = (a_n + b_n)/2$, con un error ϵ que está acotado mediante la expresión

$$\epsilon < \frac{b - a}{2^n}$$

El objetivo de este ejercicio es escribir un programa que calcule e imprima en pantalla la aproximación numérica de la raíz de una cierta función $f(x)$ obtenida por el método de bisección, con un error menor que una cierta tolerancia `Err`. El programa debe seguir el esquema indicado más abajo. La función cuya raíz queremos determinar ha sido definida previamente y devuelve el valor que toma para cada valor de x pasado como argumento de la función. Los extremos del intervalo inicial $[a, b]$ y la tolerancia del error, `Err`, deberán ser pasados por línea de comandos como argumentos de la función `main()`. Por supuesto, asumiremos que existe una única raíz de la función dentro del intervalo.

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

double funcion(double x)
{
   double f;
   ...     //f=f(x)
   return f;
}

int main(int argc, char **argv)
{
    ...

    return 0;
}
```

2. (2 puntos) Una matriz ortogonal A es una matriz cuadrada cuya matriz inversa A^{-1} coincide con su matriz traspuesta A^T: $A^{-1} = A^T$ o también $A \cdot A^T = 1$ donde 1 es la matriz identidad. Supongamos que tenemos una biblioteca de funciones en la que tenemos definida una función que calcula la matriz inversa de una matriz cuadrada. Esta función está declarada del siguiente modo

   ```
   void invierte_matriz(int orden, double A[][orden], double A_inverse[][orden])
   ```

   donde `orden` es el orden de la matriz cuadrada A (es decir, si el orden es 2 la matriz será 2×2, y `A_inverse[][]` es la matriz inversa calculada.

   Escribir una función externa a `main()` declarada del siguiente modo

   ```
   int comprueba_matriz_ortogonal (int orden, double A[][orden])
   ```

   que reciba como argumentos el orden de la matriz y la propia matriz, y que devuelva el valor 1 si la matriz introducida es ortogonal y 0 si no lo es.