MÉTODOS MATEMÁTICOS (PARTE INFORMÁTICA)

EXAMEN SEPTIEMBRE 2004

Ejercicio 1.

En el fichero de datos "`vector.txt`" se encuentran almacenadas las componentes de un vector entero **v**. Cada componente ocupa una línea distinta del fichero, es decir, hay tantos registros como componentes. El número de componentes debe ser determinado por el programa.

Se pide diseñar un programa C que realice las siguientes tareas:

- Dimensionar dinámicamente el vector **v**.
- Leer los datos del fichero y almacenarlos en **v**.
- Ordenar el vector de menor a mayor utilizando el **método de la burbuja**.
- Una vez que el vector esté ordenado, mover cada elemento una posición a la derecha, pasando la última componente al primer lugar (quedará el elemento de valor máximo en primera posición y las restantes componentes ordenadas de menor a mayor).
- Escribir en pantalla:
    - El valor máximo es: .....
    - El vector ordenado es: .....

Septiembre 2004. Solución al ejercicio 1.

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

void main( )
{
	int*v;
	int n=0,aux,i,j;
	FILE *pf;

	/*se cuenta el número de registros del fichero*/
	pf=fopen("vector.txt","r");
	while(!feof(pf))
	{
		fscanf(pf,"%d",&aux);
		n=n+1;
	}

	/*dimensionamiento dinámico*/

	v=(int*)malloc(n*sizeof(int));
	if(v==NULL)
	{
		printf("Error de localización");
		exit(0);
	}

	/*lectura de datos*/
	rewind(pf);
	for(i=0;i<n;i++) fscanf(pf,"%d",&v[i]);

	/*ordenación del vector. Método de la burbuja*/

	for(j=n-1;j>=1;j--)
	for(i=1;i<=j;i++)
	{
		if(v[i]<v[i-1])
		{
			aux=v[i];
			v[i]=v[i-1];
			v[i-1]=aux;
		}
	}


	/*se pasa cada elemento una posición a la derecha*/

	aux=v[n-1];
	for(i=n-1;i>=1;i--)v[i]=v[i-1];
	v[0]=aux;

	printf("El valor máximo es: %d\n",v[0]);
	printf("El vector ordenado es: ");
	for(i=1;i<n;i++)printf("%d ",v[i]);
	printf("%d\n",v[0]);
}
```

Ejercicio 2.

Escribir la salida a pantalla del siguiente programa C, cuando se leen por teclado los valores 12 y 360 para la variable N.

```
#include<stdio.h>
void main( )
{
        int N,i,j,f;
        printf("Introduzca un número entero \n");
        scanf("%d",&N);
        if(N<15)
        {
                for(i=1;i<=N;i++)
                {
               for(j=1;j<=i;j++)
                     printf("%d ",j);
                      printf("\n");
                }
        }
        else
        {
                f=2;
                while(f<=N)
                {
                        if(N%f==0)
                        {
                          printf("%d \n",f);
                          N=N/f;
                        }
                        else
                        {
                          f++;
                        }
                }
        }

}
```

Examen extraordinario de septiembre. Solución al ejercicio 2

Si N=12:

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

Si N=360:

```
2
2
2
3
3
5
```

Ejercicio 3.
La posición de un proyectil en tiro parabólico, en el instante t, viene dada por las ecuaciones:

$$x = ut\cos(a)\,,\quad y = ut\sin(a) - \frac{gt^2}{2}$$

Siendo u la velocidad de lanzamiento, a el ángulo de lanzamiento en grados y g la aceleración de la gravedad.

Las componentes horizontal y vertical de la velocidad son:

$$V_x = u\cos(a)\ ,\ V_y = u\sin(a) - gt$$

y su módulo $V = \sqrt{V_x^2 + V_y^2}$. El vector velocidad forma un ángulo θ con la horizontal siendo $\tan(\theta) = \frac{V_y}{V_x}$

El siguiente programa en MATLAB (**tpb1.m**), recibe como datos el ángulo de lanzamiento y el tiempo de vuelo del proyectil y calcula la posición, el módulo de la velocidad y la inclinación en ese instante de tiempo. Se pide completar las sentencias en las que aparecen las líneas de puntos.

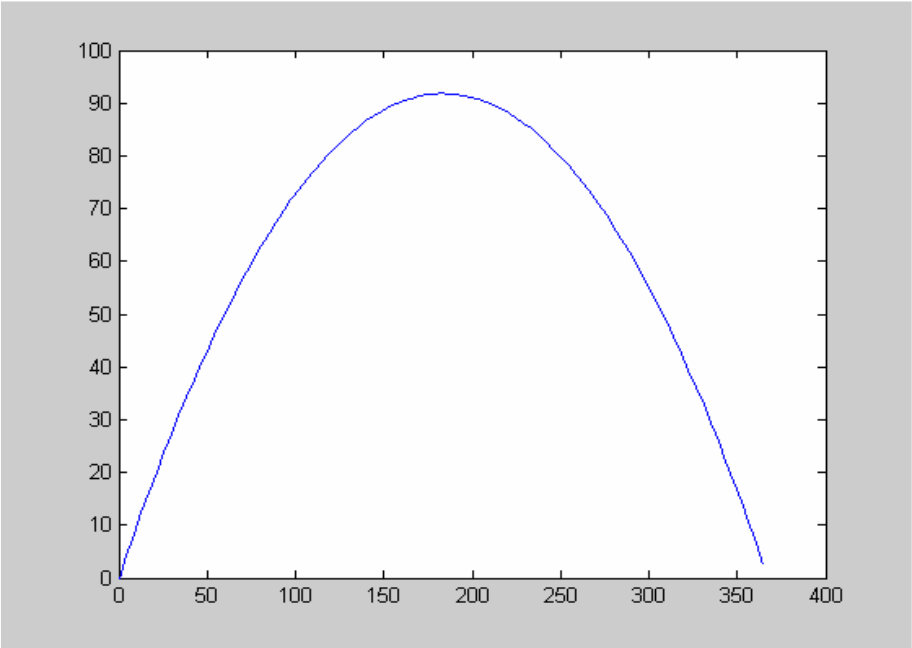
```
d= input ('Angulo de lanzamiento en grados: ');
t= input ('Tiempo de vuelo: ');
a=d*pi/180;          %conversion a radianes
u=60;                %Velocidad de lanzamiento
g=9.8;
x= ..............;                %distancia horizontal
y= ..............................; %distancia vertical
vx= ............;                  %velocidad horizontal
vy= ..................;           %velocidad vertical
V = ..................;           %velocidad resultante
th = 180 /pi * atan2( vy , vx );   %inclinacion en el instante t
disp('x:'); disp(x);
disp('y:'); disp(y);
disp('V:'); disp(V);
disp('th:');disp(th);
```

El siguiente programa **tpb2.m** calcula el alcance del proyectil. Este cálculo se efectúa incrementando la variable tiempo mientras la coordenada vertical permanezca positiva. Se pide completar el código.

```
dt = 0.1;
g = 9.8;
u = 60;
ang = input (' Angulo de lanzamiento en grados:  ');
ang = ang*pi/180;            %conversion a radianes
x = 0 ; y = 0 ; t = 0;       %condiciones iniciales
while ..............
      t=t+....;
      y = u * sin(ang) * t - g * t^2/2;
      x = u * cos(ang) * t;
end
disp('alcance:'); disp(x)
```

El siguiente programa **tpb3.m** dibuja la trayectoria del proyectil tal y como se muestra en la gráfica. Se pide completar el código.

```
dt = 0.1;
g = 9.8;
u = 60;
ang = input (' Angulo de lanzamiento en grados:  ');
ang = ang*pi/180;  %conversion a radianes
xp=......;yp=.......; %inicializacion
y=0;t = 0;
while ..........
      t=t+.....;
      y = u * sin(ang) * t - g * t^2/2;
      if .........
            x_nueva=u*cos(ang)*t;
            y_nueva=y;
            xp = ........;
            yp = ........;
      end
end
.......................
```



Examen de septiembre de 2004
Ejercicio de MATLAB (SOLUCION)
tpb1.m:

```
d= input ('Angulo de lanzamiento en grados: ');
t= input ('Tiempo de vuelo: ');
a=d*pi/180;         %conversion a radianes
u=60;               %Velocidad de lanzamiento
g=9.8;
x= u * t * cos(a);                  %distancia horizontal
y= u * t * sin(a) - 0.5 * g * t^2; %distancia vertical
vx= u * cos(a);                     %velocidad horizontal
vy= u * sin(a) - g * t;             %velocidad vertical
V = sqrt (vx^2+vy^2);               %velocidad resultante
th = 180 /pi * atan2( vy , vx );    %inclinacion en el instante t
disp('x:'); disp(x);
disp('y:'); disp(y);
disp('V:'); disp(V);
disp('th:');disp(th);
```

tpb2.m:

```
dt = 0.1;
g = 9.8;
u = 60;
ang = input (' Angulo de lanzamiento en grados:  ');
ang = ang*pi/180;             %conversion a radianes
x = 0 ; y = 0 ; t = 0;        %condiciones iniciales
while y>=0
      t=t+dt;
      y = u * sin(ang) * t - g * t^2/2;
      x = u * cos(ang) * t;
end
disp('alcance:'); disp(x)
```

tpb3.m:

```
dt = 0.1;
g = 9.8;
u = 60;
ang = input (' Angulo de lanzamiento en grados:  ');
ang = ang*pi/180;               %conversión a radianes
xp  =zeros(1);  yp  =zeros(1);  %inicialización  (también  vale
xp(1)=0,...)
y=0;t = 0;
while y>=0
t=t+dt;
y = u * sin(ang) * t - g * t^2/2;
if y>= 0
      x_nueva=u*cos(ang)*t;
      y_nueva=y;
      xp=[xp x_nueva];
      yp=[yp y_nueva];

end
end
 plot(xp, yp)
```