Apellidos: **Nombre:** **Matrícula:**

Examen

105000119 - Programación para Sistemas

10MI-Grado en Matemáticas e Informática

Lenguajes y Sistemas Informáticos e Ingeniería de Software
ETSI Informáticos
Universidad Politécnica de Madrid

Curso 2017/2018 - Julio 2018

Normas

- El examen puntúa sobre **15 puntos**.
- La duración total del mismo es de **45 minutos**.
- Se deberá tener el DNI o el carnet de la UPM en lugar visible.
- No olvidar rellenar **apellidos, nombre y número de matrícula** en cada hoja.
- La solución al examen se proporcionará antes de la revisión.

Cuestionario

(1 punto) 1. Sea el script en Bash:

```
#! /bin/bash
if cd /tmp; then
  echo 'A'
else
  echo 'B'
fi
pwd
```

Suponga que se ejecuta el script anterior en un terminal. Indique qué saldrá en pantalla.

Nota: suponga que el script es ejecutable, y que el directorio '/tmp' existe y es accesible con cd.

Solución:

A

/tmp

(1 punto) 2. Indique dos comandos básicos para mostrar manuales y documentación de ayuda de comandos Unix y Bash.

Solución: man help

(1 punto) 3. Sea el script Bash:

```
#! /bin/bash
test -d /tmp || exit 1
exit 0
```

Suponga que se ejecuta el script anterior en un terminal. Indique qué saldrá en pantalla si, después, en el mismo terminal, se ejecuta:

```
echo $?
```

Nota: suponga que el script es ejecutable, y que el directorio '/tmp' existe.

Solución: 0

(1 punto) 4. En Unix, indique la instrucción correspondiente para establecer el permiso de ejecución de un fichero miscript.sh.

Solución: chmod +x miscript.sh

(1 punto) 5. Realice un listado del directorio /usr/bin con una llamada a 'ls' de forma que envíe la salida estándar al fichero /tmp/listado1.txt, y la salida error al fichero /tmp/listado2.txt .

Solución:

```
ls /usr/bin > /tmp/listado1.txt 2> /tmp/listado2.txt
```

(1 punto) 6. Dado el siguiente código fuente de un programa en C llamado p1, escriba el resultado de la ejecución

./p1 hola.

```
#include <stdio.h>
int main( int argc, char *argv[] ){
        printf("Nombre␣del␣Programa␣%s\n", argv[0]);
        if( argc == 2 ){printf("Argumento␣%s\n", argv[1]);}
        else{printf("Faltan␣argumentos.\n");}
return 0;
}
```

Solución:

Nombre del Programa ./p1

Argumento hola

(1 punto) 7. Escriba las sentencias necesarias para declarar y reservar memoria dinámica para un vector de 200 caracteres.

Solución:

```
char *vcaracteres;
vcaracteres = malloc( 200 * sizeof(char) );
```

(1 punto) 8. Escriba en cuál de las siguientes funciones se produce reserva de memoria dinámica: calloc strcpy free malloc strdup

Solución: calloc, malloc, strdup

(1 punto) 9. Escriba las sentencias necesarias para abrir y poder escribir en un fichero llamado prueba.txt

Solución:

```
file *fp;
fp=fopen ("prueba.txt" "w");
```

(1 punto) 10. Dada la siguiente declaración de la variable libro.

```
struct Libros
{
char titulo[50];
int identificador;
} libro;
```

Escribir las sentencias que permitan guardar la información de un libro con título *Programming* e identificador 555.

Solución:

```
strcpy( libro.titulo, "Programming");
libro.identificador=555;
```

(1 punto) 11. Dado el siguiente código, escriba su salida tras su ejecución.

```
#include <stdio.h>
int main ()
{
int var[] = {10, 100, 200};
int i, *ptr;

ptr = var;
printf("Valor␣1␣=␣%d\n",*ptr );
ptr++;
printf("Valor␣2␣=␣%d\n",*(ptr+1) );

return 0;
}
```

Solución: Valor 1 = 10

Valor 2 = 200

(1 punto) 12. Dado la declaración del vector int n[10] , escribir las sentencias necesarias para inicializar cada uno de los elementos con un valor de 1.

Solución:

```
for ( i = 0; i < 10; i++ )
{
   n[ i ] =  1;
}
```

(1 punto) 13. Dado el siguiente código, escriba su salida tras su ejecución.

```
#include <stdio.h>
int main ()
{
/* local variable definition */
int a = 15;
```

```
/* while loop execution */
while( a < 18 )
{
printf("Valor de a: %d\n", a);
a++;
}
printf("Valor de a: %d\n", a);
}
```

Solución: Valor de a: 15
Valor de a: 16
Valor de a: 17
Valor de a: 18

(1 punto) 14. En lenguaje C, indique la instrucción que muestra en salida error el texto:
Mensaje de error

Solución:

```
fprintf(stderr, "Mensaje de error");
```

(1 punto) 15. Se va a utilizar la siguiente declaración de doble puntero en lenguaje C:

```
char **ppchar;
```

la cual creará una variable tipo puntero:

```
ppchar
-------
|     |-->
-------
```

Se desea llegar a obtener el siguiente diagrama:

```
ppchar
-------   -------   -----
|     |-->|     |-->|'A'|
-------   -------   -----
```

donde ppchar apunta a un puntero que a su vez apunta a un carácter al que se ha asignado el valor 'A'.

Complete el código siguiente para conseguirlo, debiéndose después liberar la memoria dinámica que se haya asignado antes de finalizar el programa.

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

int main(void) {

  char **ppchar;
  /* Inicio del c\'odigo a completar */


  /* Fin del c\'odigo a completar */
  return 0;
}
```

Solución:

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

int main( void ) {

  char **ppchar;

  ppchar = ( char ** ) malloc( sizeof( char * ) );
  if ( ppchar == NULL) { exit( 1 ); }

  *ppchar = ( char * ) malloc( sizeof( char ) );
  if ( *ppchar == NULL) { exit( 1 ); }

  **ppchar = 'A';

  /* liberar la memoria din\'amica asignada */
  free( *ppchar );
  free( ppchar );

  return 0;
}
```