PRÁCTICA OPCIÓN A

Se creará un programa por ejercicio o práctica, el nombre del programa será nombre del alumno y el número de ejercicio:

Ejemplo: Marciano\_1.

Se subirá el ejercicio al aula virtual Educamos y se grabará en el pendrive del profesor, si este lo ve conveniente realizarlo. El nombre del alumno también debe aparecer en el main como un comentario de todos los ejercicios.

|  |
| --- |
| **Resultado de aprendizaje trabajado**  **RA 1: Reconoce la estructura de un programa informático, identificando y relacionando los elementos propios del lenguaje de programación utilizado.** |
| **Criterio de Evaluación** |
| |  | | --- | | b) Se han creado proyectos de desarrollo de aplicaciones.  h) Se ha comprobado el funcionamiento de las conversiones de tipo explícitas e implícitas.  j) Se han creado y utilizado procedimientos y funciones | |

|  |
| --- |
| **Resultado de aprendizaje trabajado**  **RA 3: Escribe y depura código, analizando y utilizando las estructuras de control del lenguaje.** |
| **Criterio de Evaluación** |
| |  | | --- | | **e) Se han creado programas ejecutables utilizando diferentes estructuras de control** | |

|  |
| --- |
| **Resultado de aprendizaje trabajado**  **RA 4: Desarrolla programas organizados en clases analizando y aplicando los principios de la programación orientada a objetos.** |
| **Criterio de Evaluación** |
| |  | | --- | | **Todos** | |

Práctica 1. (10% del RA1)

1.- Crear varias funciones de conversión. La primera realizará el módulo de dos variables enteras y devuelva un tipo lógico, verdadero si el primer número es divisible por el segundo y falso, al contrario, si no es divisible, la función se debe llamar divisible. Otra función deberá devolver un número aleatorio, los argumentos de entrada serán el valor máximo y el valor mínimo de los cuales debe sacar el número aleatorio, el valor mínimo no está incluido en la selección y el máximo si estaría incluido en la selección, esta función se llamará aleatorio. Una tercera función donde se le pasen dos números y sea capaz de contar desde el menor al mayor, esta función se llamará contar.

El programa main generará dos números aleatorios entre 1 y 100, y después deberá comprobar si el primero es divisible por el segundo. Estos dos números se utilizarán para la función contar. El programa main debe de utilizar de forma coherente las funciones anteriores.

Práctica 2. (30 % RA3)

Crea una aplicación que dibuje una pirámide invertida de asteriscos. Nosotros le pasamos la altura de la pirámide por teclado, número impar. Tiene que tener control de errores (en nuestro caso probamos con 9). Este es un ejemplo:

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

Ejercicio 3. (80% del RA1) y (60% del RA3)

Una sucursal de lotería, tiene números de cinco cifras desde el 10000 al 99999. Estos números serán introducidos por teclado y se debe controlar el error, no dejando seguir el programa sino cumplen las condiciones. Una vez cumplidas las condiciones se realizarán una serie de comprobaciones siempre utilizando funciones, el número entrará en la función y esta devolverá la solución. Si el tercer número es un tres se seguirá la comprobación. Si la suma total de todos los números es múltiplo de 2 se sigue la comprobación y finalmente si el primer y último número coinciden se imprime el número por teclado. Si ha pasado todas las comprobaciones se pasará a la segunda fase, en el momento que no pase una comprobación se debe de pedir otro número por teclado y volver a empezar y seguirá en la primera fase.

En la segunda fase el programa debe de generar el número inverso al introducido imprimirlo por pantalla, después generara un número aleatorio de cinco cifras, comparara los dos números por cifras, la cifra en la posición 1 del número introducido por teclado con la cifra en la posición 1 del número aleatorio, decidirá cuál es el mayor y realizara un bucle imprimiendo el número menor tantas veces como el mayor diga, en una sola línea. Si los dos números son iguales elige una como mayor y otro como menor. Sucesivamente se comparará la 2 cifra, la 3 cifra … hasta la 5 cifra.

Se pide programa principal y funciones necesarias. No se pueden utilizar String para manejar los números.

Ejercicio 4. (80% del RA4)

Dungeons & Dragons (en español: Dragones y mazmorras o Calabozos y dragones) es un juego de rol de fantasía heroica actualmente publicado por Wizards of the Coast. El juego original fue diseñado en Estados Unidos por Gary Gygax y Dave Arneson y publicado por primera vez en 1974 por la compañía de Gygax, Tactical Studies Rules (TSR).

Originalmente derivado de juegos de tablero jugados con lápiz, papel y dados, la primera publicación de Dungeons & Dragons es bien conocida como el principio de los juegos de rol modernos​ y por consiguiente de la industria de los juegos de rol.

Este juego tiene una serie de elementos como son los dados, pueden existir dados tradicionales de seis caras hasta dados de 12 caras. Los dados decidirán en el momento de su creación el tipo máximo de caras que tendrá, después de delimitar el tamaño del dado, los dados podrán dar un resultado siempre que se tiren en el tablero.

Otro elemento que tienen el juego son las cartas, las cartas pueden ser de tres tipos Dragones, Héroes y Lugares. Una carta tiene siempre un poder y índice de vida de la carta. La carta se podrá utilizar siempre que el índice de vida de la carta sea superior a cero. El valor del índice de vida puede ir desde 5 hasta 100 puntos. El valor del poder puede valer desde 10 a 20 puntos.

Las cartas se enfrentarán entre ellas, la carta que tenga mayor poder junto con el dado lanzado ganará y se descontará de su índice de vida.

Los dados son propiedad de las cartas, cada carta tiene un solo dado, los jugadores tienen cinco cartas para empezar a jugar.

El juego empieza con dos jugadores, y sus cinco cartas para cada uno de los jugadores. Una vez repartidas las cartas, se empezará el enfrentamiento, cada jugador debe seleccionar una de sus cartas, si esa carta puede jugar, una carta podrá jugar si su índice de vida es superior a 0, la seleccionará y la enfrentará con la carta del otro jugador.

Del jugador se puede saber en todo momento la fuerza de todas sus cartas (será la suma de todos los índices de vida de sus cartas). También puede saber en todo momento el valor de todas sus cartas, será un listado de todas las cartas con sus características.

De los dos jugadores tendremos al jugador principal y al secundario. El principal es el que juega en casa y será el encargado de evaluar las tiradas y dar el resultado. El jugador principal además después de cada tirada deberá de analizar si el juego puede continuar, para ello debe de saber si siguen en juego por lo menos siete cartas, en el momento que el número de cartas con las que se puede jugar sea menor de siete cartas terminara el juego.

En este programa no se pueden utilizar listas de colecciones (arrays, matrices, conjuntos, listas, …). Se piden al menos dos constructores por clase, entre ellos en constructor sin argumentos, todos los constructores serán secundarios. Todos los objetos que se creen se realizaran a través de una factoría de clases, en alguna parte del programa se debe utilizar variables estáticas de una forma coherente, también podemos encontrar clases enumeradas.

Práctica 5. (20% RA4).

Haz una clase llamada Contraseña que siga las siguientes condiciones. Que tenga los atributos longitud, contraseña y fuerte. Por defecto, la longitud será de 8. Los constructores serán como mínimo los siguiente, un constructor por defecto. Un constructor con la longitud que nosotros le pasemos, revisa los constructores para poder tener todas las posibles posibilidades de crear contraseñas con estas características. Si la contraseña tiene activado fuerte a verdadero, las contraseñas que generen deben llevar 2 mayúsculas, 1 minúscula y 5 números, en este no tendrá en cuenta la longitud. Las letras mayúsculas que puede utilizar son [QWER], las minúsculas que utilizará son [asd] y los números [3456789]. Si fuerte no está activado, generará contraseñas de números del tamaño de longitud. Siempre que se le pida esta clase nos debe dar una contraseña. Crea un main para probarla y ver su funcionamiento completo.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
| RA1  Práctica 1 | No realiza nada | El programa no funciona y sus estructuras son erróneas y la idea de resolución es errónea y las variables y constante no son adecuadas | El programa no funciona y sus estructuras de control son erróneas, algunas variables o constantes son correctas. | El programa no funciona, la idea de resolución no es la más correcta pero se aproxima y además las estructuras no son las adecuadas | El programa funciona o funciona parcialmente pero la utilización de las estructuras son incorrectas | El programa realiza lo que se pide de una manera bastante funcional pero la utilización del código es mejorable, utiliza funciones de una manera errática pero correcta. | El programa realiza lo que se pide de una manera bastante funcional y la idea de resolución es bastante acertada pero la utilización de funciones es mejorable | El programa funciona y la utilización de las estructuras y funciones son bastante acertados | El programa está perfecto y realiza lo que se pide, pero no utiliza todas las estructuras y funciones de control necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de control y funciones necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras y funciones de control necesarias y lo realiza eficientemente, además está perfectamente documentado con nombres correctos de identificación |
| Puntos | 0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | 1 |
| RA1  Ejercicio 3 | No realiza nada | El programa no funciona y sus estructuras son erróneas y la idea de resolución es errónea y las variables y constante no son adecuadas | El programa no funciona y sus estructuras de control son erróneas, algunas variables o constantes son correctas. | El programa no funciona, la idea de resolución no es la más correcta pero se aproxima y además las estructuras no son las adecuadas | El programa funciona o funciona parcialmente pero la utilización de las estructuras son incorrectas | El programa realiza lo que se pide de una manera bastante funcional pero la utilización del código es mejorable, utiliza funciones de una manera errática pero correcta. | El programa realiza lo que se pide de una manera bastante funcional y la idea de resolución es bastante acertada pero la utilización de funciones es mejorable | El programa funciona y la utilización de las estructuras y funciones son bastante acertados | El programa está perfecto y realiza lo que se pide, pero no utiliza todas las estructuras y funciones de control necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de control y funciones necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras y funciones de control necesarias y lo realiza eficientemente, además está perfectamente documentado con nombres correctos de identificación |
|  | 0 | 0.5 | 1 | 2.5 | 4 | 4.5 | 5 | 5.5 | 6 | 7 | 8 |
| RA3  Práctica 2 | No realiza nada | El programa no funciona y sus estructuras de control son erróneas y la idea de resolución es errónea | El programa no funciona y sus estructuras de control son erróneas | El programa no funciona, la idea de resolución no es la más correcta pero se aproxima y además las estructuras no son las adecuadas | El programa funciona o funciona parcialmente pero la utilización de las estructuras son incorrectas | El programa realiza lo que se pide de una manera bastante funcional pero la utilización del código es mejorable | El programa realiza lo que se pide de una manera bastante funcional y la idea de resolución es bastante acertada pero la utilización del código es mejorable | El programa funciona y la utilización de las estructuras son bastante acertados | El programa está perfecto y realiza lo que se pide, pero no utiliza todas las estructuras de control necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de control necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de control necesarias y lo realiza eficientemente, además está perfectamente documentado con nombres correctos de identificación |
| Puntos | 0 | 0.2 | 0.3 | 0.5 | 0.7 | 0.8 | 1 | 1.5 | 2 | 2.5 | 3 |
| RA3  Ejercicio 3 | No realiza nada | El programa no funciona y sus estructuras de control son erróneas y la idea de resolución es errónea | El programa no funciona y sus estructuras de control son erróneas | El programa no funciona, la idea de resolución no es la más correcta pero se aproxima y además las estructuras no son las adecuadas | El programa funciona o funciona parcialmente pero la utilización de las estructuras son incorrectas | El programa realiza lo que se pide de una manera bastante funcional pero la utilización del código es mejorable | El programa realiza lo que se pide de una manera bastante funcional y la idea de resolución es bastante acertada pero la utilización del código es mejorable | El programa funciona y la utilización de las estructuras son bastante acertados | El programa está perfecto y realiza lo que se pide, pero no utiliza todas las estructuras de control necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de control necesarias | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de control necesarias y lo realiza eficientemente, además está perfectamente documentado con nombres correctos de identificación |
| Puntos | 0 | 0.5 | 1 | 2.5 | 3 | 3.5 | 4 | 4.5 | 5 | 5.5 | 6 |
| RA4  Ejercicio 4 | No realiza nada | El programa no funciona y sus estructuras de control son erróneas y la idea de resolución de clases es errónea | El programa no funciona y las clases son erróneas | El programa no funciona, la idea de resolución no es la más correcta pero se aproxima y además las clases no son las adecuadas, ni los atributos | El programa funciona o funciona parcialmente pero la utilización de las clases son incorrectas | El programa realiza lo que se pide de una manera bastante funcional pero las clases pueden ser mejorables | El programa realiza lo que se pide de una manera bastante funcional y la idea de resolución de clases es bastante acertada pero no es del todo correcta | El programa funciona y la utilización de las clases son bastante acertados | El programa está perfecto y realiza lo que se pide, pero no utiliza todas la potencia de la programación orientada a objetos | El programa está perfecto y realiza lo que se pide, utiliza todas las clases con sus atributos y métodos. | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de clases necesarias y lo realiza eficientemente, además está perfectamente documentado con nombres correctos de identificación y utiliza todos los elementos pedidos de estructuras de clases. |
|  | 0 | 0.5 | 1 | 2 | 3.5 | 3 | 4 | 5 | 6 | 7 | 8 |
| RA4  Práctica 5 | No realiza nada | El programa no funciona y sus estructuras de control son erróneas y la idea de resolución de clases es errónea | El programa no funciona y las clases son erróneas | El programa no funciona, la idea de resolución no es la más correcta pero se aproxima y además las clases no son las adecuadas, ni los atributos | El programa funciona o funciona parcialmente pero la utilización de las clases son incorrectas | El programa realiza lo que se pide de una manera bastante funcional pero las clases pueden ser mejorables | El programa realiza lo que se pide de una manera bastante funcional y la idea de resolución de clases es bastante acertada pero no es del todo correcta | El programa funciona y la utilización de las clases son bastante acertados | El programa está perfecto y realiza lo que se pide, pero no utiliza todas la potencia de la programación orientada a objetos | El programa está perfecto y realiza lo que se pide, utiliza todas las clases con sus atributos y métodos. | El programa está perfecto y realiza lo que se pide, utiliza todas las estructuras de clases necesarias y lo realiza eficientemente, además está perfectamente documentado con nombres correctos de identificación y utiliza todos los elementos pedidos de estructuras de clases. |
|  | 0 | 0.5 | 0.8 | 0.9 | 1 | 1.2 | 1.4 | 1.6 | 1.7 | 1.8 | 2 |