07/20

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
EXAMEN DE: FUNDAMENTOS DE PROGRAMACIÓN
FORMA A PRACTICA

PERÍODO: 2025-S01 PARCIAL: 3
NOMBRE: Gosdillo Monserrate CURSO (NRC): 24583
CARRERA: Electronica y Automatizacion FECHA: 21/01/2026

INDICACIONES GENERALES:

- La evaluación es personal y no se permite intercambiar información con sus compañeros, si es llamado la atención por una vez, se procederá a ponerle CERO y no recibirle su evaluación.
- La parte práctica debe subirle En la carpeta de GitHub y formato indicado por el Sr. Docente (Apellidos Nombres_U2_Examen),
- **Dispone de 2h.**

Parte Práctica (20 Puntos)
Instrucciones.

1. A continuación, se presentan los requisitos funcionales (RF) para el desarrollo de su programa
2. Recuerde que en el código elaborado por Ud. Debe comentar donde desarrolla el RF solicitado , en este documento coloque el código y al final las capturas de pantalla que evidencien los RF.
3. Desde ya éxitos en su evaluación.
4. Este documento debe ser subido en formato PDF con Apellidos Nombres_U2_Examen, a la tarea al AULA VIRTUAL

1. REQUISITOS FUNCIONALES

RF01 – Configuración dinámica del rango: El programa debe permitir que el usuario defina el rango mínimo y máximo del número secreto.
RF02 – Configuración dinámica de intentos: El programa debe permitir que el usuario seleccione el número máximo de intentos (hasta 10).
RF03 – Validación de rango y duplicados: El programa debe validar que cada intento esté dentro del rango y no se repita.
RF04 – Registro de intentos en matriz: El programa debe almacenar en una matriz el número de intento, el valor ingresado y el resultado (0 bajo, 1 alto, 2 correcto).
RF05 – Visualización de resumen detallado: Al terminar, el programa debe mostrar una tabla con cada intento y el resultado textual, y revelar el número secreto si no se adivinó.

2. RUBRICA DE CALIFICACIÒN

Criterio	4 puntos – Excelente	3 puntos – Bueno	2 puntos – Aceptable	1 punto – Deficiente	EVALUACION
Inicialización y generación del número aleatorio.	Número aleatorio generado correctamente y una sola vez.	Se genera bien, pero fuera de lugar lógico.	Se generan varios números o el rango es incorrecto.	No se genera el número correctamente.	0
Ingreso y validación de datos	Captura todos los intentos y valida correctamente.	Captura intentos, pero tiene validaciones limitadas.	Captura intentos, pero con errores menores.	No captura correctamente los intentos.	2

$)dx =$

$ormula$

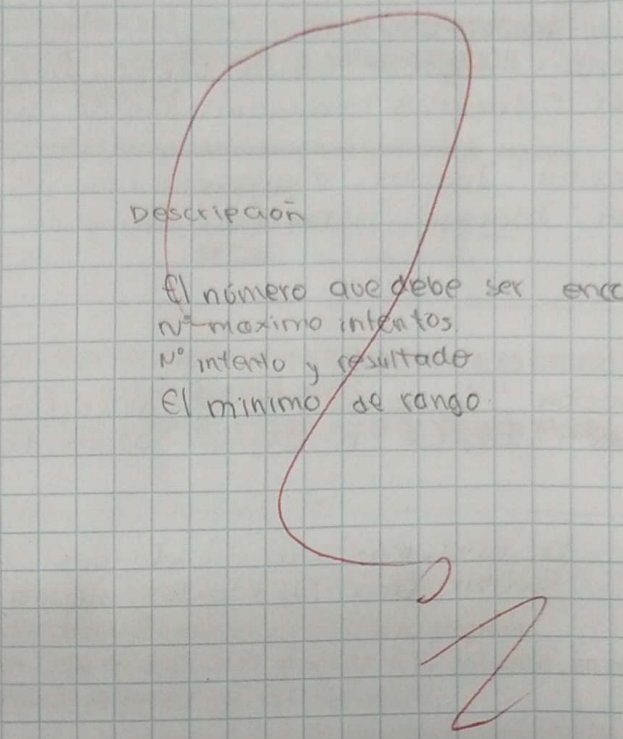
Uso de la matriz para almacenar datos	Matriz implementada correctamente y se imprime al final.	Matriz implementada, pero con errores menores.	Uso parcial de la matriz.	No se usa la matriz adecuadamente.	1
Condiciones y control de flujo	Comparación exacta, con mensajes adecuados y flujo lógico.	Mensajes adecuados con mínimo error de flujo.	Comparación básica con mensajes genéricos.	Lógica confusa o incorrecta.	1
Mensajes finales y condición secreta	Mensaje secreto visible solo si acierta, resumen completo.	Muestra el resumen, pero con fallos en el mensaje.	Muestra el mensaje incluso cuando no acierta.	No se muestran resultados correctamente.	3
				TOTAL, SOBRE 20 PUNTOS	

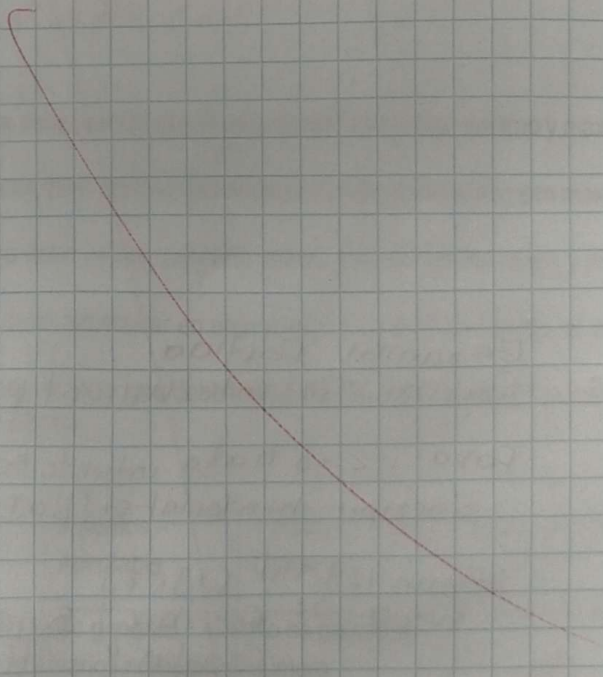
Elaborado por: Ing Jenny A Ruiz R
Docente TC DCCO
Fecha:21/01/2026

Nombre: Gordillo Monserrate
NRC: 29583
Fecha: 21/01/2026

Tabla de objetos

Tipo	Nombre	Descripción
entero	num_sec	El número que debe ser encontrado
constante	num_max	N° máximo intentos
entero	num_int	N° intento y resultado
entero	num_min	El mínimo de rango





```
Psent
   IrAcaAlgoritmo
   Escribir "Ingrese numero minimo de rango"
   Leer " Ingrese numerominimo de rango"

   Escribir "Ingrese numero maximo de rango"
   Leer "Ingrese numero maximo de rango"



   Escribir "Ingres N. intentos desea tener"
   Leer "Ingrese N. intentos desea tener"

      Si num_max <10 entonces
      Escribir Erro: N. intentos alcanzados
```



```
      Resumen Partida
RQUS  Escribir "Intento | Valor | Resultado

      Para i <- 0 hasta intentoActual
         Escribir historial[i][0], " | " historial[i][1]

      segun historial[i][2]
         hacer 0: Escribir "bajo " 0
                  Escribir "  alto " 1
                  Escribir "  correcto 2

      Fin Segun.
      Fin Para

      Si gano = Falso
         Escribir "Se agoto el intento
         El numero era :", numSecreto
      Fin Si

      Fin.
```

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

NOMBRE: Monserrate Gordillo
NRC:
FECHA : 23/01/2025
ASIGNATURA: Fundamentos de la programación
CARRERA: Ingeniería Electrónica y Automatización
TEMA: corrección evaluación

Link del GDB OBLINE

https://onlinegdb.com/wIAJ7La4a

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

#define MAX_INTENTOS 10
#define TAM_TEXTO    30

typedef struct {
    int valor;
    char resultado[TAM_TEXTO];  // "Correcto", "Menor que el secreto", "Mayor que el
secreto"
} Intento;

// --- Prototipos ---
int leerEntero(const char *mensaje);
int leerEnteroEnRango(const char *mensaje, int min, int max);
int esDuplicado(const Intento intentos[], int usados, int valor);
void imprimirResumen(int min, int max, int secreto, const Intento intentos[], int usados,
int maxIntentos);

int main(void) {
    int min, max;
    int maxIntentos;
    int numeroSecreto;
    Intento intentos[MAX_INTENTOS];
    int usados = 0;

    // ---------------------------
    // RF01: Configuración del rango
    // ---------------------------
    printf("=== CONFIGURACION DEL RANGO ===\n");
    min = leerEntero("Ingrese el minimo del rango: ");
    max = leerEntero("Ingrese el maximo del rango: ");
```

```
    if (min >= max) {
        printf("ERROR: El minimo debe ser menor que el maximo.\n");
        return 1;
    }

    // Inicializar RNG y secreto
    srand((unsigned)time(NULL));
    numeroSecreto = (rand() % (max - min + 1)) + min;

    // ---------------------------
    // RF02: Configuración de intentos
    // ---------------------------
    printf("\n=== CONFIGURACION DE INTENTOS ===\n");
    maxIntentos = leerEnteroEnRango("Seleccione el maximo de intentos (1 a 10): ", 1,
MAX_INTENTOS);

    printf("\n=== COMIENZA EL JUEGO ===\n");

    // ---------------------------
    //  RF03: VALIDACION DE RANGO Y DUPLICADOS
   // ---------------------------
    while (usados < maxIntentos) {
        int valor;
        printf("\nIntento %d de %d. Ingrese un numero: ", usados + 1, maxIntentos);
        if (scanf("%d", &valor) != 1) {
            // Limpiar buffer en caso de entrada no numérica
            int c; while ((c = getchar()) != '\n' && c != EOF) {}
            printf("ERROR: Entrada no valida. Intente nuevamente.\n");
            continue;
        }

        // Validar rango (RF03)
        if (valor < min || valor > max) {
            printf("ERROR: Numero fuera del rango [%d - %d]\n", min, max);
            continue;
        }

        // Validar duplicado (RF03)
        if (esDuplicado(intentos, usados, valor)) {
            printf("ERROR: Ya ingresaste este numero antes.\n");
            continue;
        }
         // ---------------------------
         //RF04:REGISTRO DE INTENTO DE MATRIZ
         // ---------------------------
        intentos[usados].valor = valor;

        // Calcular resultado textual para RF05
        if (valor == numeroSecreto) {
```

```
            strncpy(intentos[usados].resultado, "Correcto", TAM_TEXTO - 1);
            intentos[usados].resultado[TAM_TEXTO - 1] = '\0';
        } else if (valor < numeroSecreto) {
            strncpy(intentos[usados].resultado, "Menor que el secreto", TAM_TEXTO - 1);
            intentos[usados].resultado[TAM_TEXTO - 1] = '\0';
        } else {
            strncpy(intentos[usados].resultado, "Mayor que el secreto", TAM_TEXTO - 1);
            intentos[usados].resultado[TAM_TEXTO - 1] = '\0';
        }


        if (valor == numeroSecreto) {
            printf("¡FELICIDADES! Adivinaste el numero secreto.\n");
            usados++;  // se cuenta el intento ganador
            break;
        } else if (valor < numeroSecreto) {
            printf("El numero secreto es MAYOR.\n");
        } else {
            printf("El numero secreto es MENOR.\n");
        }

        usados++;
    }

    // ---------------------------
    // RF05: Resumen detallado en tabla
    // ---------------------------
    imprimirResumen(min, max, numeroSecreto, intentos, usados, maxIntentos);

    if (usados == maxIntentos && (usados == 0 || intentos[usados - 1].valor !=
numeroSecreto)) {
        printf("\nNo lograste adivinar el numero. ¡Sigue intentando!\n");
    }

    return 0;
}


int leerEntero(const char *mensaje) {
    int v;
    for (;;) {
        printf("%s", mensaje);
        if (scanf("%d", &v) == 1) {
            return v;
        }
        // limpiar buffer
        int c; while ((c = getchar()) != '\n' && c != EOF) {}
        printf("Entrada no valida. Intente nuevamente.\n");
    }
}
```

```
int leerEnteroEnRango(const char *mensaje, int min, int max) {
    int v;
    for (;;) {
        printf("%s", mensaje);
        if (scanf("%d", &v) == 1 && v >= min && v <= max) {
            return v;
        }
        int c; while ((c = getchar()) != '\n' && c != EOF) {}
        printf("ERROR: Debe estar entre %d y %d.\n", min, max);
    }
}

int esDuplicado(const Intento intentos[], int usados, int valor) {
    for (int i = 0; i < usados; ++i) {
        if (intentos[i].valor == valor) {
            return 1;
        }
    }
    return 0;
}

void imprimirResumen(int min, int max, int secreto, const Intento intentos[], int usados,
int maxIntentos) {
    printf("\n==========================================\n");
    printf(" RESUMEN DETALLADO DEL JUEGO\n");
    printf("==========================================\n");
    printf("Rango elegido: [%d - %d]\n", min, max);
    printf("Numero secreto: %d\n", secreto);
    printf("Intentos permitidos: %d\n", maxIntentos);
    printf("Intentos realizados: %d\n", usados);
    printf("Intentos restantes: %d\n\n", maxIntentos - usados);

    // Tabla
    printf("-----------------------------------------------------------\n");
    printf("%-10s %-15s %-20s\n", "Intento", "Valor", "Resultado");
    printf("-----------------------------------------------------------\n");
    for (int i = 0; i < usados; ++i) {
        printf("%-10d %-15d %-20s\n", i + 1, intentos[i].valor, intentos[i].resultado);
    }
    printf("-----------------------------------------------------------\n");
}
```