

Nº Parcial:

Nombre:

CI:

Primer Parcial. Programación 1

Instituto de Computación

Curso 2022

Observaciones:

- El parcial consta de 10 preguntas y un ejercicio de programación.
- Cada pregunta tiene una única opción correcta.
- Una respuesta correcta suma 3 puntos. Una respuesta incorrecta resta 0.75 puntos. Una pregunta sin respuesta no afecta el puntaje.
- El ejercicio de programación tiene un total de 10 puntos.
- Debe entregar la planilla con las respuestas junto con la letra y la resolución del ejercicio. Todo debe estar identificado con nombre y cédula.
- En todos los ejercicios se utiliza el lenguaje Pascal tal como fue dado en el curso (ej: circuito corto, case con else, etc.).

Ejercicio 1 Dadas las siguientes declaraciones

```
const M = ... ; {un entero mayor que 1}
type arreglo = array [1 .. M] of integer;
var arr : arreglo;
    i : integer;
```

Indique cuál de los siguientes fragmentos de código imprime correctamente *exito* si hay dos elementos consecutivos iguales en el arreglo *arr* y *fracaso* en caso contrario:

a)

```
i := 1;
while (i < M) and (arr[i] <> arr [i+1]) do
    i := i+1;
if arr[i] = arr [i+1] then
    writeln('exito')
else writeln('fracaso');
```

b)

```
i := 1;
while (i < M) and (arr[i] <> arr [i+1]) do
    i := i+1;
if i = M then
    writeln('fracaso')
else writeln('exito');
```

c)

```
i := 1;
while (arr[i] <> arr [i+1]) and (i <= M) do
    i := i+1;
if i <= M then
    writeln('exito')
else writeln('fracaso');
```

d)

```
i := 1;
while (i <= M) and (arr[i] <> arr [i+1]) do
    i := i+1;
if i <= M then
    writeln('exito')
else writeln('fracaso');
```

e)

```
i := 1;
while (i < M) and (arr[i] <> arr [i+1]) do
    i := i+1;
if i <= M then
    writeln ('exito')
else writeln ('fracaso');
```

Ejercicio 2 Dada la siguiente declaración de variables, ¿cuál asignación **NO** produce un error de compilación?

```
VAR n : Integer;
a,b : Char;
```

- a) `n := ord(1.5)`
- b) `n := chr('1')`
- c) `n := ord(a) div ord(b)`
- d) `'a' := b`
- e) `a := b + '1'`

Ejercicio 3 Dada la siguiente declaración de subprograma:

```
PROCEDURE Parcial (valorReal: Real;
                    valorEntero: Integer;
                    VAR Resultado: Real);
```

¿Cuál de las siguientes invocaciones **NO** es correcta? (donde *num* es de tipo *Integer* y *arg* es de tipo *Real*) Todas las variables han sido correctamente inicializadas.

- a) `Parcial (num, 3, arg);`
- b) `Parcial (5*num, trunc (6.2), arg);`
- c) `Parcial (arg, 3, arg);`
- d) `Parcial (num, 3, 2.0);`
- e) `Parcial (1, 3, arg);`

Ejercicio 4 Dado el siguiente programa, indique cuál de las siguientes afirmaciones es correcta:

```
program e1;
var i,a,b : integer;
begin
    a := 4;
    for i := 1 to 5 do
        begin
            b := i;
            while (b < a) and (a-b > 2) do
                b := b + 1;
            if b < a then
                write('WOW', ' ');
            else write(i * b, ' ');
        end
    end
end.
```

- a) el programa escribe 1 4 9 16 25
- b) el programa escribe WOW WOW WOW WOW 25
- c) el programa escribe WOW WOW 9 16 25
- d) el programa escribe WOW WOW WOW 16 25
- e) el programa escribe WOW WOW WOW WOW WOW

Ejercicio 5 Dado el siguiente programa:

```
PROGRAM Ejercicio (output);
VAR entero, valor : Integer;

PROCEDURE pascal (entero : Integer;
                  VAR resultado : Integer);
VAR valor : Integer;
BEGIN
    valor := 2 * entero + resultado;
    entero := entero + valor;
    resultado := resultado + entero;
END;

BEGIN
    entero := 1;
    valor := 0;
    pascal (entero, valor);
    writeln (entero, valor);
    valor := valor + 1;
    pascal (valor, entero);
    writeln (entero, valor)
END.
```

La salida correcta es:

- a) 1 3
14 4
- b) 1 3
1 4
- c) 1 0
5 3
- d) 1 2
3 3
- e) 1 2
14 3

Ejercicio 6 Dadas las siguientes instrucciones

```
A) while condicion1 do
    cuerpo1

B) repeat
    cuerpo2
until condicion2
```

¿Cuál de las siguientes afirmaciones **NO** es correcta?

- a) cuerpo1 puede no ejecutarse nunca.
- b) A termina cuando condicion1 es falsa.
- c) B termina cuando condicion2 es verdadera.
- d) cuerpo2 se ejecuta al menos una vez.
- e) cuerpo1 se ejecuta al menos una vez.

Ejercicio 7 Dadas las siguientes declaraciones:

```
CONST N = ...; {un entero mayor que 0}
TYPE Arreglo = ARRAY[1..N] OF Integer;
VAR b : Arreglo; i : Integer;
```

Indicar cuál de los siguientes códigos inicializa correctamente un arreglo de enteros de modo que las celdas de índice impar contengan el entero 1 y las otras contengan el entero 0. La constante N puede ser par o impar.

a)

```
FOR i := 1 TO N DIV 2 DO
BEGIN
    b[i*2] := 0;
    b[i*2+1] := 1
END
```

b)

```
FOR i := 1 TO N DO
    if i mod 2 = 0 then
        b[i] := 0
    else
        b[i] := b[i-1] + 1
```

c)

```
FOR i := 1 TO N DIV 2 DO
BEGIN
    b[i*2] := 0;
    b[i*2-1] := 1
END
```

d)

```
FOR i:= 1 TO N DO
    b[i] := i mod 2
```

e)

```
FOR i:= 1 TO N DIV 2 DO
    b[i*2] := 0
```

Ejercicio 8 Dado el siguiente programa, ¿cuál afirmación es correcta?

```
program iter(input,output);
var x,i: integer;
begin
x:= 1;
for i:= 5 downto 1 do
    x:= i * x;
write(x);
end.
```

- a) el programa no compila
- b) el programa produce un error de ejecución
- c) el programa imprime 120
- d) el programa imprime 5 20 60 120 120
- e) el programa no imprime nada

Ejercicio 9 Indique cuál fragmento de código lee un número entero en la variable num y escribe solamente *insuficiente* si está entre 0 y 3, *suficiente* si está entre 4 y 10 y *fuera de rango* en otro caso.

a)

```
readln (num);
if (num <= 10) and (num >= 0) then
    case num of
        0, 1, 2, 3 :
            writeln ('insuficiente')
        else writeln ('suficiente')
    end
else writeln ('fuera de rango')
```

b)

```
readln (num);
if (num <= 10) then
    if (num >= 0) then
        case num of
            0, 1, 2, 3 :
                writeln ('insuficiente')
        end
    else writeln ('suficiente');
writeln ('fuera de rango')
```

c)

```
readln (num);
if (num <= 10) then
    if (num >= 0) then
        case num of
            0, 1, 2, 3 :
                writeln ('insuficiente')
        end
    else writeln ('suficiente')
else writeln ('fuera de rango')
```

d)

```
readln (num);
if (num <= 10) and (num >= 0) then
begin
    case num of
        0, 1, 2, 3 :
            writeln ('insuficiente')
    end;
    writeln ('suficiente')
end;
writeln ('fuera de rango')
```

e)

```
readln (num);
if (num <= 10) then
    if (num >= 0) then
        case num of
            0, 1, 2, 3 :
                writeln ('insuficiente')
            else writeln ('suficiente')
        end
    else writeln ('fuera de rango')
```

Ejercicio 10 Dado el siguiente programa:

```
program ej;
var i : integer;
    cond : boolean;
begin
    readln(i) ;
    cond := true;
    while (i < 5) or not cond do
    begin
        repeat
            i := (i + 1) mod 10
        until i mod 2 = 0;
        cond := not (i mod 2 = 0)
    end;
    if cond then
        writeln(i)
end.
```

Indique la opción correcta

- a) para cualquier valor de i el programa termina y no imprime nada
- b) para i = 0 el programa imprime 5
- c) para cualquier valor de i el programa no termina
- d) para cualquier valor de i entre 0 y 4 el programa no termina
- e) el programa da error en tiempo de ejecución

Ejercicio de Resolución

- El puntaje máximo por este ejercicio es **10 puntos** (no resta puntos).
- Este ejercicio debe resolverse en la carátula.
- En este ejercicio se evaluará, además de la lógica correcta, la utilización de un buen estilo de programación de acuerdo a los criterios impartidos en el curso. De esta manera se tendrá en cuenta entre otros conceptos: indentación apropiada, correcta utilización de las estructuras de control, código elegante y legible, eficiencia de los algoritmos, etcétera.

Dadas las siguientes declaraciones:

```
const N = ...; (* N entero > 1 *)
type arreglo = array [1..N] OF 1 .. M (* M > 0 *);
```

Escriba la función:

```
function poseeSumaPrecede (a : Arreglo) : boolean;;
```

que dado un arreglo, determina si posee alguna celda cuyo valor sea igual a la suma de todos los valores almacenados en las celdas anteriores a ella.

Ejemplos para N = 5:

a	Resultado
[7,5,3,2,1]	FALSE
[1,2,1,2,1]	FALSE
[1,1,8,6,4]	TRUE
[1,2,3,6,4]	TRUE
[1,1,1,1,4]	TRUE