

Nº Parcial:  
Nombre:  
CI:

- a) 11
- b) 45
- c) 10
- d) 9
- e) 55

## Primer Parcial. Programación 1

Instituto de Computación  
Curso 2019

Respuesta al final: <sup>3</sup>

Ejercicio 4 Dado el siguiente procedimiento:

```
procedure Proc;  
var a, b: integer;  
  procedure p (x: integer; var y: integer);  
  begin  
    x := x + 1;  
    y := y + 1;  
    write(x,y);  
  end;  
begin  
  a := 1;  
  b := 0;  
  p(b, a);  
  writeln(a,b);  
end;
```

¿qué se despliega al ejecutarlo?

- a) 1210
- b) 1212
- c) 1220
- d) 2110
- e) 2121

Respuesta al final: <sup>4</sup>

Ejercicio 5 Dado el siguiente fragmento de código en el que a y b son variables de tipo integer:

```
if not (a > 5) and (b > 5) then  
  result := 1  
else  
  case a of  
    6: result := 1  
  else  
    result := -1  
  end;  
writeln(result);
```

¿para cuál asignación de las variables a y b se despliega -1?

- a) a := 4; b := 5;
- b) a := 5; b := 6;
- c) a := 6; b := 4;
- d) a := 5; b := 7;
- e) a := 4; b := 6;

Respuesta al final: <sup>5</sup>

Ejercicio 6 ¿Cuál de los siguientes subrangos NO es correcto?

- a) 1.0 .. 1.9
- b) 0 .. maxint
- c) false .. true
- d) 'a' .. 'a'
- e) 'A' .. 'Z'

Respuesta al final: <sup>6</sup>

Ejercicio 7 ¿Cuál es el resultado de la ejecución del siguiente fragmento de código en el que la variable i es de tipo integer?:

Nº Parcial:  
Nombre:  
CI:

### Observaciones:

- El parcial consta de 16 preguntas y un ejercicio de programación.
- Cada pregunta tiene una única opción correcta.
- Una respuesta correcta suma 2 puntos. Una respuesta incorrecta resta 0.5 puntos. Una pregunta sin respuesta no afecta el puntaje.
- El ejercicio de programación tiene un total de 8 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 Dado el siguiente fragmento de código en el que x e y son de tipo integer:

```
x := 1;  
y := 4;  
while (x <= 4) or (y > 2) do  
begin  
  if x <= 4 then  
  begin  
    x := x + 1;  
    write(x)  
  end  
  else  
  begin  
    y := y - 2;  
    write(y)  
  end  
end;
```

¿cuál de estas afirmaciones es correcta?

- a) El programa imprime 234520.
- b) El programa imprime 23452.
- c) El programa imprime 123452.
- d) El programa imprime 1234520.
- e) La iteración es infinita.

Respuesta al final: <sup>1</sup>

Ejercicio 2 ¿Cuál de las siguientes expresiones tiene error de tipos?

- a) 7 + trunc(9.2)
- b) ord('A') \* 6
- c) 3 + 5.1
- d) 5 + 'A'
- e) 2.3e-1 / 3

Respuesta al final: <sup>2</sup>

Ejercicio 3 Dado el siguiente fragmento de código:

```
suma := 0;  
i := 0;  
repeat  
  i := i + 1;  
  suma := suma + i  
until suma > 50;
```

¿cuál es el valor de i al terminar la iteración?

```

i := 0;
if (i = 0) or (10 div i = 0) then
  write(i)
else
  write(i + 1);

writeln('-');

```

- a) Si se evalúa por circuito completo se despliega 1-.
- b) Si se evalúa por circuito completo se despliega -.
- c) Si se evalúa por circuito corto da error en tiempo de ejecución.
- d) Si se evalúa por circuito corto se despliega 0-.
- e) Si se evalúa por circuito corto se despliega 1-.

**Respuesta al final:** <sup>7</sup>

**Ejercicio 8** Dados los siguientes procedimientos:

```

procedure externo(x:integer; var t:integer);
var a, b, c: integer;

  procedure interno1(x, t:integer);
  var a, z: integer;
  begin
    (* interno1 *)
  end;

  procedure interno2(var a:integer);
  var v, w: integer;

    procedure masinterno(var x, t:integer);
    var a: integer;
    begin
      (* masinterno *)
    end;

  begin
    (* interno2 *)
  end;

begin
  (* externo *)
end;

```

¿cuál de estas afirmaciones es correcta?

- a) En el procedimiento `interno2` son visibles las variables locales `v`, `w` y las variables `a`, `x`, `t` del procedimiento `masinterno`.
- b) En el procedimiento `interno1` solamente son visibles las variables locales `x`, `t`, `a`, `z`.
- c) En el procedimiento `masinterno` solamente son visibles las variables locales `x`, `t`, `a`, y las variables `v`, `w` del procedimiento `interno2`.
- d) En el procedimiento `externo` son visibles las variables locales `x`, `t`, `a`, `b`, `c`, la variable `z` del procedimiento `interno1` y las variables `v`, `w` del procedimiento `interno2`.
- e) En los procedimientos `interno1`, `interno2` y `masinterno` son visibles las variables `b` y `c` del procedimiento `externo`.

**Respuesta al final:** <sup>8</sup>

**Ejercicio 9** Dado el siguiente programa:

```

program ejercicio;
var
  num: array [1..8] of integer;
  i, a, y: integer;
begin
  num[1] := 1;
  for i := 2 to 8 do
    num[i] := num[i-1]*2;
  readln(a);
  y := 0;
  for i := 1 to 2 do
    y := y + num[a*i];
  write(y)
end.

```

¿cuál de estas afirmaciones es correcta?

- a) Si `a` es 4 se imprime 80.
- b) Si `a` es 4 se imprime 136.
- c) Si `a` es 2 se imprime 20.
- d) Si `a` es 2 se imprime 8.
- e) Si `a` es 4 se produce error en tiempo de ejecución.

**Respuesta al final:** <sup>9</sup>

**Ejercicio 10** Dado el siguiente programa:

```
program ej;
var
  c : char;
  i : integer;
begin
  i := 1;
  read(c);
  while (c < 'J') or (i <= 4) do
  begin
    i := i + 1;
    read(c)
  end;
  writeln(c, ' ', i)
end.
```

¿qué se despliega al recibir como entrada AXBYCZD?

- a) Y 4
- b) X 2
- c) Z 6
- d) Y 3
- e) Z 5

**Respuesta al final:** <sup>10</sup>

**Ejercicio 11** La expresión booleana  $\text{not } p \text{ or } q \text{ and } p$ , al ser evaluada, siempre produce el mismo valor que la expresión booleana:

- a)  $(\text{not } p) \text{ or } (q \text{ and } p)$
- b)  $((\text{not } p) \text{ or } q) \text{ and } p$
- c)  $(\text{not } (p \text{ or } q)) \text{ and } p$
- d)  $\text{not } ((p \text{ or } q) \text{ and } p)$
- e)  $\text{not } (p \text{ or } (q \text{ and } p))$

**Respuesta al final:** <sup>11</sup>

**Ejercicio 12** Dadas las siguientes declaraciones:

```
type arr = array [1..9] of boolean;
var a : arr;
```

¿cuál asignación es correcta?

- a)  $a[\text{true}] := \text{false}$
- b)  $a[3] := a[0] \text{ and } a[5]$
- c)  $\text{arr}[1] := \text{not } \text{arr}[8]$
- d)  $a[9] := a[1] \text{ and } \text{true}$
- e)  $a[1..9] := \text{false}$

**Respuesta al final:** <sup>12</sup>

**Ejercicio 13** Dada la siguiente instrucción en la que  $v$  es una variable de tipo real:

```
v := round (15.8 - 0.4 / 2) div 2 / 2
```

¿cuál de estas afirmaciones es correcta?

- a) La variable  $v$  toma el valor 2.0.
- b) La variable  $v$  toma el valor 16.0.
- c) La variable  $v$  toma el valor 4.0.
- d) La variable  $v$  toma el valor 3.5.
- e) La variable  $v$  no toma ningún valor ya que la expresión no compila.

**Respuesta al final:** <sup>13</sup>

**Ejercicio 14** Dado el siguiente fragmento de código:

```
for i := n downto 0 do
  write('*');
```

¿cuál de estas afirmaciones es correcta?

- a) Si  $n \geq 0$  se despliegan  $n$  asteriscos.
- b) Si  $n < 0$  se produce un error en tiempo de ejecución.
- c) Si  $n < 0$  la ejecución del ciclo no termina nunca.
- d) Si  $n < 0$  no se despliega nada y no se produce ningún error.
- e) Si  $n = 0$  no se despliega nada y no se produce ningún error.

**Respuesta al final:** <sup>14</sup>

**Ejercicio 15** Dadas la siguientes definiciones:

```
const M = (* valor entero mayor que 1 *)
type arreglo = array [1..M] of integer;
function pertenece (x: integer; A: arreglo): boolean;
(* retorna true si x pertenece al arreglo A *)
var i: integer;
<BLOQUE>
```

y los siguientes opciones para <BLOQUE>:

**código 1**

```
begin
  i:= 1;
  while (i <= M) and (A[i] <> x) do
    i:= i + 1;
  pertenece:= i <= M
end; {pertenece}
```

**código 2**

```
begin
  i:= 1;
  while (i < M) and (A[i] <> x) do
    i:= i + 1;
  pertenece:= A[i] = x
end; {pertenece}
```

**código 3**

```
begin
  i:= 1;
  while (A[i] <> x) and (i <= M) do
    i:= i + 1;
  pertenece:= i <= M
end; {pertenece}
```

**código 4**

```
begin
  i:= 1;
  while (A[i] <> x) and (i <= M) do
    i:= i + 1;
  pertenece:= A[i] = x
end; {pertenece}
```

**código 5**

```
begin
  i:= 0;
  while (A[i] <> x) and (i < M) do
    i:= i + 1;
  pertenece:= A[i] = x
end; {pertenece}
```

¿cuál de estas afirmaciones es correcta?

- a) El código 1 **NO** da error si se evalúa con circuito corto.
- b) El código 2 da error si se evalúa por circuito completo.
- c) El código 3 **NO** da error si se evalúa con circuito corto.
- d) El código 4 solo da error si se evalúa por circuito completo
- e) El código 5 **NO** da error si se evalúa por circuito corto o completo.

**Respuesta al final:** <sup>15</sup>

**Ejercicio 16** Dadas las siguientes definiciones:

```
var a, b: integer;
procedure procl(VAR par: integer);
begin
  par := 1;
  a := par;
end;

function funl(pf1: integer): integer;
begin
  funl := pf1 + 1;
end;
```

¿con cuál de estos fragmentos de código se despliega 2?

- a) procl(funl(1)); writeln(a);
- b) procl(2); writeln(a);
- c) procl(b); funl(a); writeln(a);
- d) procl(b); writeln(funl(a));
- e) a := 1; procl(funl(a)); writeln(a);

**Respuesta al final:** <sup>16</sup>

## Ejercicio de Resolución

- El puntaje máximo por este ejercicio es **8 puntos** (no resta puntos).
- Este ejercicio debe resolverse en esta hoja.
- 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.

1. Escriba un subprograma `multDato`

```
function multDato(num1, num2: integer): boolean;
```

que, dados dos números enteros mayores que 0, determine si el primero es múltiplo del segundo.

Ejemplos:

```
multDato (40, 4) es true
multDato (25, 3) es false
```

2. Escriba un programa que lea un dato entero mayor que 0 y una secuencia de números enteros mayores que 0 y despliegue la cantidad de números de la secuencia que son múltiplos del dato, usando el subprograma de la parte 1). Los enteros de la secuencia son ingresados separados por espacios y siempre se ingresará el valor 0, para indicar el fin de la entrada.

Ejemplos:

Dato: 3	Dato: 4	Dato: 2	Dato: 4
Secuencia: 333 25 84 14 0	Secuencia: 33 56 80 15 22 4 0	Secuencia: 1 3 5 0	Secuencia: 0
Salida: 2	Salida: 3	Salida: 0	Salida:

### Respuesta

## Solución:

Versión en que si la secuencia es vacía se despliega 0.

```
program ejerCodParcial2019;
var
  cant, dato, num: integer;

  function multDato(num1, num2: integer): boolean;
  begin
    multDato := (num1 mod num2 = 0)
  end;

begin
  readln(dato);
  read(num);
  cant := 0;
  while num > 0 do
  begin
    if multDato(num, dato) then cant := cant + 1;
    read(num)
  end;
  writeln(cant)
end.
```

Versión en que si la secuencia es vacía no se despliega nada.

```
program ejerCodParcial2019;
var
  cant, dato, num: integer;

  function multDato(num1, num2: integer): boolean;
  begin
    multDato := (num1 mod num2 = 0)
  end;

begin
  readln(dato);
  read(num);
  if num > 0 then
  begin
    cant := 0;
    repeat
      if multDato(num, dato) then cant := cant + 1;
      read(num)
    until num = 0
    writeln(cant)
  end
end.
```

## Respuestas

<sup>1</sup>b

<sup>2</sup>d

<sup>3</sup>c

<sup>4</sup>c

<sup>5</sup>a

<sup>6</sup>a

<sup>7</sup>d

<sup>8</sup>e

<sup>9</sup>b

<sup>10</sup>c

<sup>11</sup>a

<sup>12</sup>d

<sup>13</sup>c

<sup>14</sup>d

<sup>15</sup>a

<sup>16</sup>d