

Nº Parcial:

Nombre:

CI:

Primer Parcial. Programación 1

Instituto de Computación

Setiembre 2022

Observaciones:

- El parcial consta de 14 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 12 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 código:

```
program secuencia;  
var  
  c : char;  
  cont : integer;  
begin  
  cont := 0;  
  read(c);  
  repeat  
    if (c >= 'a') and (c <= 'z') then  
      cont := cont + 1;  
    read(c)  
  until c = '$';  
  write(cont)  
end.
```

- El programa despliega la cantidad de letras minúsculas leídas, si la entrada es una secuencia no vacía de caracteres terminada con el centinela '\$'.
- El programa da error de compilación porque no es posible utilizar operadores relacionales con valores de tipo `char`.
- Si el símbolo '\$' no es ingresado en la entrada el programa igual despliega la cantidad de letras minúsculas leídas.
- El programa despliega la cantidad de letras minúsculas leídas, si la entrada es una secuencia de caracteres, que puede ser vacía, terminada con el centinela '\$'.
- El programa despliega la cantidad de caracteres leídos, si la entrada es una secuencia no vacía de caracteres terminada con el centinela '\$'.

Ejercicio 2 Dado el siguiente programa:

```
program distanciaPuntos;  
  
var abs1, abs2, orde1, orde2 : real;  
  
function distancia(x1, x2, y1, y2 : real) : real;  
begin  
  distancia := sqrt(sqr(x2 - x1) + sqr(y2 - y1))  
end;  
  
begin  
  writeln('Ingrese coordenadas primer punto: ');  
  readln(abs1, orde1);  
  writeln('Ingrese coordenadas segundo punto: ');  
  readln(abs2, orde2);  
  writeln('La distancia entre los puntos es: ');  
  distancia(abs1, abs2, orde1, orde2);  
end.
```

¿Cuál de las siguientes opciones es correcta?

- El programa despliega la distancia entre los puntos de coordenadas ingresadas.
- La función calcula mal la distancia entre los puntos de coordenadas ingresadas.
- La función está mal invocada.
- Los valores de las coordenadas deben leerse dentro de la función.
- Falta una instrucción en la función para desplegar el resultado.

Ejercicio 3 Dado el siguiente programa:

```
program ejercicio;  
  
var x, y : integer;  
  
begin  
  read(x);  
  y := 0;  
  while (x <> 0) and (y div x = 0) do  
    begin  
      if x > 0 then  
        begin  
          x := x - 1;  
          y := y + 1  
        end;  
      if x < 0 then x := -x  
    end;  
    write(x + y)  
  end.
```

Cual de las siguientes afirmaciones es correcta:

- El programa da error en tiempo de ejecución a causa de una división por cero.
- La ejecución del `while` nunca termina, para algún valor leído.
- El programa despliega el valor absoluto del valor leído.
- El cuerpo del `while` nunca se ejecuta, cualquiera sea el valor leído.
- Si el valor leído es negativo, el cuerpo del `while` se ejecuta una sola vez.

Ejercicio 4 Dadas las declaraciones:

```
var
  existe, divide : boolean;
  i, j           : integer;
```

Indique cuál de las siguientes asignaciones compila correctamente.

- a) `divide := not divide or existe j > 8`
- b) `divide := not divide and not (j = i + 1)`
- c) `i := i + j > 10`
- d) `trunc(j) := i div j`
- e) `existe := (not j) + i <= 0`

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

```
if a > b then p(a-1, b) else p(b-1, a);
```

¿Cuál de los siguientes cabecales es correcto?

- a) `procedure p(a : real; var b : integer);`
- b) `procedure p(var a : integer; b : integer);`
- c) `procedure p(var a : integer; b : real);`
- d) `procedure p(a : integer; var b : real);`
- e) `function p(a, b : integer);`

Ejercicio 6 Dado el siguiente programa:

```
program p;
var n, cont : integer;
begin
  cont := 0;
  read(n);
  while (cont < 1000) or (n > 0) do
  begin
    n := n - 1;
    cont := cont + 1
  end;
  write(cont)
end.
```

Indicar cuál afirmación es correcta.

- a) El cuerpo del `while` no se ejecuta nunca si el valor leído es negativo.
- b) El cuerpo del `while` no termina nunca si el valor leído es negativo.
- c) El cuerpo del `while` no termina nunca, cualquiera sea el valor leído.
- d) El programa despliega 1000, si el valor leído es menor o igual que 1000, o `n`, si el valor leído es mayor que 1000.
- e) El programa siempre despliega 1000.

Ejercicio 7 Dado el siguiente encabezado de un procedimiento:

```
procedure calculo(
  a, b : integer;
  var c : integer;
  d : integer);
```

¿Cuál de las siguientes afirmaciones es correcta?

- a) Este procedimiento recibe cuatro parámetros por referencia.
- b) Este procedimiento recibe cuatro parámetros por valor.
- c) El cabezal de este procedimiento se puede simplificar definiendo los cuatro parámetros juntos: `(a, b, c, d : integer)`.
- d) Este procedimiento recibe dos variables por valor y dos por referencia.
- e) Este procedimiento puede recibir y/o retornar un dato usando el parámetro `c`.

Ejercicio 8 Considere que `x` es una variable real con valor mayor a cero. ¿Cuál es la expresión con menor valor de las siguientes?

- a) `trunc(x) + trunc(x)`
- b) `trunc(x + x)`
- c) `x + trunc(x)`
- d) `trunc(trunc(x + x))`
- e) `trunc(trunc(x)) + x`

Ejercicio 9 La función potencia calcula la potencia de una base `b` elevada a un exponente `e`, siendo `b` y `e` dos enteros mayores o iguales que 0. Indique qué opción debe ser colocada en donde dice OPCIÓN para obtener un resultado correcto en cualquier caso (asuma que no va a haber problemas de rango por exceder MAXINT).

```
function potencia(base, exponente : integer): integer;
var i, pot : integer;
begin
  OPCIÓN
end;
```

- a) `pot := 1;`
`for i := 2 to exponente do pot := pot * base;`
`potencia := pot`
- b) `pot := 0;`
`for i := 2 to exponente do pot := pot * base;`
`potencia := pot`
- c) `pot := base;`
`for i := 2 to exponente do pot := pot * base;`
`potencia := pot`
- d) `pot := 1;`
`for i := 1 to exponente do pot := pot * base;`
`potencia := pot`
- e) `pot := base;`
`for i := 1 to exponente do pot := pot * base;`
`potencia := pot`

Ejercicio 10 Indicar cuál es la salida del siguiente programa:

```
program pp;
var x, a, p : integer;
begin
  x := 2;
  p := 0;
  for a := 1 to 4 do p := p * x + a;
  writeln(p)
end.
```

- a) 42
- b) 0
- c) 36
- d) 26
- e) 64

Ejercicio 11 Dado el siguiente fragmento de código en el que *a* y *b* son variables de tipo *integer*, y *exp* una expresión booleana.

```
a := b;
if exp then a := a + 1
else
  begin
    b := b + 1;
    if exp then b := b + 1 else a := a + 1;
  end
```

Indique la expresión *exp* que hace que al final de la ejecución las variables *a* y *b* tengan el mismo valor.

- a) $a = b$
- b) $a <> b$
- c) $a < b$
- d) `true`
- e) `false`

Ejercicio 12 Dado el siguiente fragmento de código:

```
read(a);
if a mod 2 = 0 then
  a := a div 2
else
  a := a * 2;
write(a);
```

¿Cuál de las siguientes opciones es correcta?

- a) Siempre que se ingresa un número par, se despliega un número impar.
- b) Siempre que se ingresa un número impar, se despliega un número par.
- c) Se despliega el mismo valor que el ingresado.
- d) Al terminar la ejecución del fragmento el valor de *a* es par siempre.
- e) Al terminar la ejecución del fragmento el valor de *a* es impar siempre.

Ejercicio 13 Dado el siguiente fragmento de programa:

```
k := 0;
for i := 1 to 6 do
  begin
    if i < 3 then k := k + 1
    else if i < 4 then k := k + 2;
  end
```

Indique el valor de *k*.

- a) 3
- b) 4
- c) 5
- d) 6
- e) No puede conocerse.

Ejercicio 14 Dado el siguiente fragmento de programa, en donde *a* es una variable de tipo *integer*:

```
read(a);
case a of
  1: write(a);
  2: write(a - 1);
  3: write(a - 2);
else
  write(a - a + 1)
end;
```

Indique qué opción es correcta:

- a) El programa siempre despliega el valor 1.
- b) El programa siempre despliega el valor $a - 1$, para cualquier valor ingresado para *a*.
- c) El programa no despliega nada si se ingresa un valor negativo.
- d) El programa despliega el valor 1 cuando se ingresa un valor mayor o igual a 0 y un valor negativo cuando se ingresa un valor negativo.
- e) El programa no despliega nada si se ingresa el valor 0.

Ejercicio de Resolución

- El puntaje máximo por este ejercicio es **12 puntos** (no resta puntos).
- 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.

a) Escribir la función:

```
function cantDivisores(num : integer) : integer;
```

que devuelve la cantidad de divisores que tiene el número num, sin contar el 1 ni el propio num. Asuma que num es mayor que cero.

Respuesta:

```
function cantDivisores(num : integer) : integer;
var i, cant : integer;
begin
  cant := 0;
  for i := 2 to num div 2 do
    if num mod i = 0 then
      cant := cant + 1;
  cantDivisores := cant;
end;
```

Solución alternativa más eficiente:

En cantDivisores se recorren todos los candidatos a ser divisores uno por uno. Sin embargo, podemos detectar dos divisores cada vez en vez de uno, ya que si $\text{num mod } i = 0$ tendremos necesariamente $\text{num mod } k = 0$ para algún k tal que $i * k = \text{num}$. Debemos cuidarnos del caso en que $i = k$ en el que no se agregan dos divisores, sino uno solo. Estas consideraciones llevan al siguiente código.

```
function cantDivisores(num : integer) : integer;
var i, cant, m : integer;
begin
  cant := 0;
  if num > 1 then
    begin
      m := trunc(sqrt(num));
      for i := 2 to m do
        if num mod i = 0 then
          cant := cant + 2;
      if m * m = num then cant := cant - 1;
    end;
  cantDivisores := cant;
end;
```

b) Escribir una programa principal que lea una secuencia de números positivos de la entrada estándar (el fin de la entrada se indica con el centinela -1) y despliegue la cantidad de divisores que tiene cada uno.

Por ejemplo, si se ingresa la secuencia 12 5 21 2 -1 el programa debe desplegar:

```
El número 12 tiene 4 divisores
El número 5 tiene 0 divisores
El número 21 tiene 2 divisores
El número 2 tiene 0 divisores
```

Respuesta:

```
program divisores;
var num : integer;

function cantDivisores(num : integer) : integer;
{ ...usa definición de la parte a)... }

begin
  read(num);
  while num <> -1 do
    begin
      writeln(num, ' tiene ', cantDivisores(num), ' divisores. ');
      read(num);
    end;
end.
```

Respuestas

1a

2c

3c

4b

5a

6d

7e

8a

9d

10d

11e

12b

13b

14a