Universidad Nacional de Río Cuarto
Facultad de Cs. Exactas, Fco-Qcas y Naturales - Departamento de Computación
Asignatura: **INTRODUCCIÓN A LA ALGORÍTMICA Y PROGRAMACIÓN**
Año: 2022

Práctica Nº 6

Tema: Acciones

Esta práctica tiene como objetivos

- Resolver problemas que requieran la utilización de acciones e identificar sus ventajas y limitaciones.
- Desarrollar acciones autocontenidas.
- Emplear los distintos tipos de pasaje de parámetros analizando ventajas y desventaja
- Utilizar tipos simples y el tipo compuesto registro.
- Resolver problemas que requieran la utilización de la composición iterativa, secuencial, condicional y funciones.
- Realizar pruebas de escritorio sencillas en los algoritmos que utilizan acciones.
- Implementar en C algoritmos que contengan acciones.
- Realizar pruebas sencillas en los programas.

Ejercicios propuestos

1) Dada la siguiente acción:

```
Acción SumRes(dato a,b ∈ Z, dato-resultado c ∈ Z, resultado multi ∈ R)
Inicio
 a ← a+1
 b ← b-5
 multi ← a*b
 si multi>=0  entonces
   c ← c + multi
 sino
   c ← c - multi
 fsi
Faccion
```

¿Cuál es la salida de los parámetros actuales después de las siguientes invocaciones a la acción SumRes, en los algoritmos que se muestran abajo?

```
Algoritmo Ejemplo1   //resuelto para que sirva de modelo
Léxico
  x, y, z ∈ Z
  p ∈ R
Acción SumRes(dato a,b ∈ Z, dato-resultado c ∈ Z, resultado multi ∈ R)
Inicio
  a ← a+1
  b ← b-5
  multi ← a*b
  si multi>=0  entonces
    c ← c + multi
  sino
    c ← c - multi
  fsi
Faccion
Inicio
  x ← 16
```

```
  y ← 5
  z ← 20
  p ← 42.0
  {e₀: x=16, y=5, z=20, p= 42 }
  SumRes(x,y,z,p)
  Salida:x y z p
  {eᶠ:x=16, y=5, z= 20, p= 0 }
Fin

Algoritmo Ejemplo2
Léxico
  x, y, z ∈ Z
  p ∈ R
Acción SumRes(dato a,b ∈ Z, dato-resultado c ∈ Z, resultado multi ∈
R)
Inicio
  a ← a+1
  b ← b-5
  multi ← a*b
  si multi>=0  entonces
    c ← c + multi
  sino
    c ← c - multi
  fsi
Faccion
Inicio
  x ← 5
  y ← 4
  z ← 0
  p ← 0.0
  {e₀:                                                    }
  SumRes(x,y,z,p)
  Salida:x y z p
  {eᶠ:                                                     }
Fin


Algoritmo Ejemplo3
Léxico
  a, b, c ∈ Z
  pr ∈ R
Acción SumRes(dato a,b ∈ Z, dato-resultado c ∈ Z, resultado multi ∈
R)
Inicio
  a ← a+1
  b ← b-5
  multi ← a*b
  si multi>=0  entonces
    c ← c + multi
  sino
    c ← c - multi
  fsi
Faccion
Inicio
  a ← 8
```

```
  b ← -6
  c ← 10
  {e◦:                                                    }
  SumRes(b,a,c,pr)
  Salida:b a c pr
  {ef:                                                    }
Fin


Algoritmo Ejemplo4
Léxico
  x, z ∈ Z
  b ∈ R
Acción SumRes(dato a,b ∈ Z, dato-resultado c ∈ Z, resultado multi ∈
R)
Inicio
  a ← a+1
  b ← b-5
  multi ← a*b
  si multi>=0  entonces
    c ← c + multi
  sino
    c ← c - multi
  fsi
Faccion
Inicio
  x ← 0
  z ←10
  b ← 100.0
  {e◦:                                                    }
  SumRes(1,z,x,b)
  Salida:z x b
  {ef:                                                    }
Fin
```

2) Dado el siguiente algoritmo:

```
Algoritmo AreaFiguras
Lexico
  x, y, z, sup ∈ R

Acción Cargar(              a, b ∈ R,          :f ∈ Caracter)
Inicio
   //ingrese una t si es un triángulo y una r si es un rectángulo
   Entrada:f
   si f= 'r'    entonces
      //ingrese el 1er  y 2do lado
      Entrada:a  b
    sino
      // ingrese la base del triángulo y la altura del triángulo
      Entrada:a  b
   fsi
Faccion

Acción Calcular(            a, b ∈ R,          f ∈ Caracter;          area ∈ R)
Lexico Local
 s ∈ R
```

```
Inicio
 si f= 'r'  entonces
    area ← a * b
 sino
    area ← a * b /2
 fsi
Faccion

Accion Mostrar (          a, b ∈ R,          f  ∈ Caracter,          area ∈ R)
Lexico local
 msge ∈ Cadena
Inicio
 Si f= 'r'  Entonces
    msge←"El área del rectángulo dado por los lados"
 Sino
   msge←"El área del triángulo dado por la altura y la base"
 fsi
 Salida:msge a b area
Faccion

Inicio //programa principal
  Cargar (x,y,z)
  Calcular(x,y,z,sup)
  Mostrar (x,y,z,sup)
Fin
```

Determine el tipo de pasaje de parámetros (dato, dato-resultado, resultado) que corresponde dar a las variables declaradas como parámetros formales en las acciones Cargar, Calcular y Mostrar.

3) a) Analice y describa lo que hace el siguiente algoritmo, y en base a ello, determine el tipo de pasaje de parámetros de cada una de las acciones que componen al mismo.
b) Desarrolla la acción MostrarMonto que corresponde al algoritmo.

```
Algoritmo CalcularPagoFormadePago
Léxico
 Tinteres=(3..15)
 apellidoNombres ∈ Cadena      // dato para almacenar el nombre del cliente
 montoCompra ∈ R       // dato para almacenar la compra $ del cliente
 esContado ∈ Lógico    // dato que permite identificar la forma de pago
 interesTarjeta ∈ Tinteres  // dato para almacenar el interés de la tarjeta
 Acción ObtenerDatos (          apeNombres ∈ Cadena, monto ∈ R, contado ∈ Lógico)
 //hacer

 Acción PagoContado (          monto ∈ R)
 Inicio
  si monto>=1000   entonces
     monto←monto-(monto*15/100)
 sino
    monto←monto-(monto*10/100)
  fsi
Faccion

 Acción IdentificaTarj(          interTarj ∈ Tinteres)
 Léxico local
   cod ∈ 1..4 //variable para determinar la tarjeta utilizada
 Inicio
  //Ingrese el número de la tarjeta utilizada -1,2,3 o 4-
  //1.Master – 2.Visa – 3.Cabal – 4.Cordobesa
   Entrada:cod
   segun
```

```
    (cod=1):interTarj←10
    (cod=2):interTarj←5
    (cod=3):interTarj←15
    (cod=4):interTarj←3
  fsegun
Faccion

Acción PagoTarjeta (                    monto ∈ R,        interTarj ∈ Tinteres)
Inicio
  monto←monto+monto*interTarj/100
Faccion

Acción MostrarMonto (                   apeNombres ∈ Cadena, monto ∈ R)
//hacer

Inicio  //del algoritmo
   ObtenerDatos(apellidoNombres, montoCompra, esContado)
   si esContado  entonces //esContado=Verdadero
      PagoContado(montoCompra)
   sino //esContado=Falso
      IdentificaTarj(interesTarjeta)
      PagoTarjeta(montoCompra, interesTarjeta)
   fsi
   MostrarMonto(apellidoNombres, montoCompra)
Fin
```

4) Desarrolla una acción que simule una calculadora. Debe recibir dos números que serán los operandos y un carácter que será el operador. En una variable resultado se almacenará el resultado de aplicar el operador a los operandos. Las operaciones que debe soportar son: '+', '-', '/', '*'. En el caso que se intente la división por cero, la acción emitirá un mensaje 'ERROR' y en la variable resultado almacenará un 999999999.

5) Con el objeto de modularizar un algoritmo más complejo se requiere una acción que permita cargar las coordenadas de un punto del plano cartesiano XY. Utilizar Tpunto=<x,y ∈ *R*>

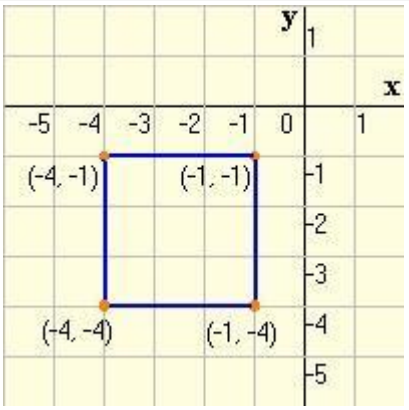
6) Con el objeto de modularizar un algoritmo más complejo se requiere una acción que permita cargar los coeficientes **a** y **b** de una recta cuando es expresada en su forma explícita: **a**x+**b**= y. Utilizar Trecta=<a,b ∈ *R*>

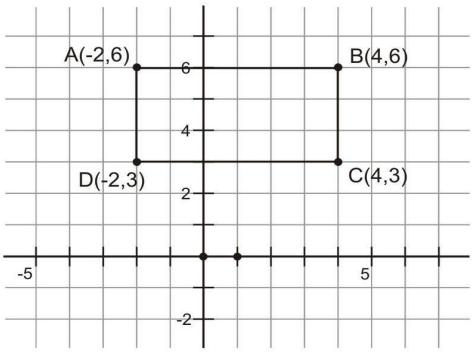
7) Con el objeto de modularizar un algoritmo más complejo se requiere una acción que reciba las coordenadas cartesianas del centro de un círculo (usar Tpunto=<x, y ∈ *R* >) y su radio.

8) Reutilice las acciones desarrolladas en ejercicios anteriores y desarrolle una nueva acción que reciba como parámetros de entrada un punto del plano cartesiano (usar Tpunto=<x, y ∈ *R* >), los coeficientes **a** y **b** de una recta (Utilizar Trecta=<a,b ∈ *R>)* cuando está expresada en la forma explícita (**a**x + **b** = y) y le asigne a una variable (que será un parámetro de resultado) valor verdadero si el punto pertenece a la recta y sino le asignará valor falso.

9) Dado un punto del plano cartesiano y las coordenadas del centro de un círculo y su radio, utilice esta información y asigne a una variable (que será un parámetro de resultado), valor verdadero si el punto está adentro del círculo y sino le asignará el valor falso. Reusar las acciones creadas en ej 5) y ej. 7) de esta práctica.

10) Desarrollar una acción que permita ingresar dos pares de coordenadas cartesianas. Las mismas representan los vértices opuestos de un rectángulo en el plano cartesiano. Utilizando esos mismos parámetros que se usan para el ingreso de los puntos, los empleará como resultado para almacenar en ellos: en el primer par la esquina inferior izquierda del rectángulo y como segundo par, la esquina superior derecha. Tener en consideración que los puntos dados como datos pueden ser de dos cualquiera de esquinas opuestas en diagonal, del rectángulo (por ejemplo puede venir como dato primero el punto A y luego el C, o primero el B y como segundo el D, etcétera). Reusar las acciones creadas en puntos anteriores que les puedan ser útiles.





11) Resolver mediante una acción el siguiente problema: se reciben como datos de entrada tres pares de coordenadas cartesianas, se debe asumir que los dos primeros puntos corresponden a esquinas opuestas de un rectángulo (pero no se sabe cuáles). El tercer par corresponde a un punto del plano cartesiano. Con esta información la acción debe asignar a una variable que será un parámetro de resultado, el valor verdadero si el punto está adentro del rectángulo y sino le asignará el valor falso. Reutilizar las acciones creadas en puntos anteriores que les puedan ser útiles.

Nota: para realizar 11 suponga ya definidas las acciones necesarias pedidas en los puntos anteriores, invocarlas con los parámetros adecuados. UTILIZAR tipo compuesto.

12) Resolver el problema planteado en el ejercicio 17 del TP N° 3 utilizando las acciones desarrolladas en los puntos anteriores. (punto dentro fuera de círculo y/o rectángulo). (ver ej 9 de esta práctica)

I) PREGUNTA: ¿Cómo se hace desde una acción para modificar un valor de una variable que está en el léxico (variable global) sin perder la característica de ser autocontenida?.

II) PREGUNTA: Dentro del cuerpo de un algoritmo, ¿donde se definen las acciones?, ¿dónde se usan las acciones?

13) a) ¿Qué hace esta acción?

Acción swap (datos-resultado x,y ∈ Z)
Inicio
x <-- x + y
y <-- x - y
x <-- x - y
Faccion

b) ¿Qué Salida daría esta acción si en un algoritmo la invocamos por ejemplo así?

Algoritmo Intercambiar
Léxico

```
 a ∈ Z
 Acción Swap (datos-resultado x,y ∈ Z)
 Inicio
    x <-- x + y
    y <-- x - y
    x <-- x - y
  Faccion
Inicio
 Entrada: a b
 Swap (a, b);
 Salida: a b
 Entrada: a
 Swap (a, a);
 Salida:a
Fin
```

14) ¿Que hace esta acción implementada en lenguaje C?

```
void Swap (int *x,int *y) {
   *x = (*x)+(*y);
   *y = (*x)-(*y);
   *x = (*x)-(*y);
}
```

Nota: que resultado mostraría cada uno de los printf de a, b, a, en el siguiente programa C:

```
#include <stdio.h>
int a,b;
void Swap (int *x,int *y) {
   *x = (*x)+(*y);
   *y = (*x)-(*y);
   *x = (*x)-(*y);
}
int main() {
 printf("introduce el valor de la variable a ");
 scanf("%i",&a);
 printf("introduce el valor de la variable b ");
 scanf("%i",&b);
 Swap (&a, &b);
 printf ("el valor de a es %d ", a);
 printf ("el valor de b es %d ", b);
 printf("\n introduce el valor de la variable a ");
 scanf("%i",&a);
 printf("introduce el valor de la variable b ");
 scanf("%i",&b);
 Swap (&a, &a);
 printf ("el valor de a es %d ", a);
 return 0;
}
```

¿Por qué el último prinf muestra ese valor de a?

15) Reescribe las acciones (incluso puedes modificar el nombre original por otro más abreviado) que se listan a continuación utilizando parámetros, considera al elegir los mismos que deben permitir a la acción ser reutilizada y presta especial atención a que los cálculos interiores a la acción se realicen sólo con los datos que recibe como parámetros:

a) Acción CalculaSalOrdinario, ver ejercicio 14) de la práctica 3.
b) Acción CalculaSalExtra, ver ejercicio 14) de la práctica 3.
c) Acción Semáforo, recibe como parámetro un color: rojo, verde o amarillo y debe devolver un parámetro de tipo adecuado, respectivamente: ‘Alto’, ‘Ádelante’ o ‘Precaución’
e) Acción CondiciónAlumno, recibe tres notas y con ellas la acción calcula la menor de las tres notas y

el promedio de las tres notas, con estos resultados informa en un parámetro adecuado la condición del alumno que será: ‘Libre’ si la menor nota es inferior a 4. ‘Regular’ si la menor nota es mayor o igual a 4 y menor a 6 y el promedio menor a 7. Por último la condición será ‘Promocionado’ si la menor nota es mayor o igual a 6 y el promedio es igual o mayor a 7. Para ver cuánto vale la menor de las notas, para calcular los promedios, debe usar funciones adecuadamente definidas. Puede utilizar las funciones creadas en la Práctica Nro 4

Plan de clases:
Clase 1: hacer el 4)
Clase 2: hacer 5) 8) y 11)
Clase 3: hacer 14) y 15.c)