UPC

PROGRAMACIÓ
Grau en Estadística UB-UPC, febrer 2021
Prof Robert Joan-Arinyo

Funcions

Les funcions en la programació de computadors són bàsicament la traducció algorísmica de les funcions de la matemàtica de manera que la diferència més notable entre la matemàtica i la programació no és tant conceptual com de notació.

1 Funcions en la matemàtica

Tot estudiant que segueixi estudis universitaris de caire tecnico-científic ha de manegar el concepte de funció matemàtica amb soltura. Per tant, el que descrivim tot seguit no pretén ser res més que un petit recordatori que farà més senzill entendre el concepte de funció en la programació de computadors.

1.1 Definició

Una *funció*, també congeuda com *aplicació* o com *transformació*, és una relació entre els elements d'un conjunt donat A, anomenat espai de definció, i els elements d'un altre conjunt B, anomenat espai imatge, de manera que a cada element a d'A fa correspondre un únic element del espai imatge b de B. La notació usual de la matemàtica d'una funció f és

$$f : A \longrightarrow B$$

Normalment, hom escriu $f(a) = b$, on b és l'únic element del conjunt B generat per f quan s'aplica a a.

Habitualment, el terme *funció* s'utilitza quan l'espai imatge és un conjunt de valors numèrics, reals o complexos. Llavors, es parla de *funció real* o *funció complexa*. Les funcions entre conjunts qualssevol s'anomenen *aplicacions*.

A tall d'exemple, la funció

$$\begin{array}{rccc} f(x,y): & \mathbb{R}^2 & \longrightarrow & \mathbb{R}^3 \\ & (x,y) & \longrightarrow & (x, y, \frac{x^2+y^2}{2p}) \end{array}$$

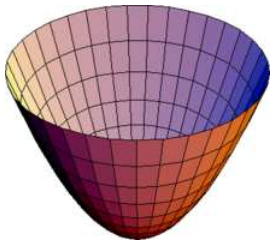

Figura 1: Paraboloide de revolució.

en donar valors reals arbitràris als paràmetres x i y, amb (x, y) de l'espai $\mathbb{R}^2$, genera el punt $(x, y, \frac{x^2+y^2}{2p})$ en $\mathbb{R}^3$. La funció defineix un paraboloide de revolució semblant al de la Figura 1.

1.2 Utilització

Com sabem, la utilització de funcions en matemàtiques és ben senzilla. Sigui $f(x, y, z)$ una funció definida en $\mathbb{R}^3$. Donat un punt de l'espai de definició, per exemple (a, b, c), només cal substituir els paràmetres simbòlics (les lletres) pels valors corresponents del punt: on hi ha una x poseu a, on hi ha una y poseu b i on hi ha una z poseu c. D'aquest procediment de substitució hom en diu *correspondència posicional* segons l'ordre dels eixos coordenats donat en la definició de la funció.

Considerem ara la funció del paraboloide definida més amunt. Si l'apliquem en el punt concret $(3, -2)$, el punt del paraboloide serà

$$f(3,\ -2) = (3,\ -2,\ \frac{3^2 + (-2)^2}{2p})$$

Notis la correspondència posicional aplicada entre els noms dels paràmetres en la definició de la funció i la substitució pels valors concrets del punt on s'aplica. Si la mateixa funció s'aplica ara al punt $(-2, 3)$, el punt resultant del paraboloide és

$$f(-2,\ 3) = (-2,\ 3,\ \frac{(-2)^2 + 3^2}{2p})$$

clarament diferent del punt del cas anterior.

2 Funcions en la programació de computadors

De manera anàloga al cas de l'ús de les funcions en matemàtiques, quan s'apliquen funcions en programació de computadors cal considerar dos aspectes clarament diferenciats: la definició i la utilització.

2.1 Definició

Una funció del llenguatge de programació **R** és un programa en general parametritzat tal que:

1. Forma part d'un altre programa.
2. S'identifica amb un nom propi únic dins del programa del qual en forma part.
3. Resol un subproblema del problema més general resolt pel programa del qual la funció forma part.
4. Rep informació via paràmetres.
5. Proporciona resultats.
6. Les variables que puguin aparèixer en la funció no tenen cap relació amb les variables dels programes dels quals la funció en forma part.

La sintaxi de definició d'una funció **R** és

```
nomDeLaFuncio <- function(p1, p2, ..., pn){

  Cos de la funcio

  return (expressio)
}
```

La primera línia de la funció

```
nomDeLaFuncio <- function(p1, p2, ..., pn){
```

s'anomena *capçalera*,o *definició* perquè defineix completament la manera en la que s'utilitzarà la funció. Les variables `p1, p2, ...,pn`, s'anomenen *paràmetres* de la funció. És possible que la llista dels paràmetres sigui buida.

En el cos de la funció hom hi programa la solució del problema corresponent a resoldre. En general, els càlculs de la funció dependran dels valors dels paràmetres.

Si, com és usual, la funció cal que retorni algun valor resultat, caldrà que hi hagi, si més no, una sentència

```
return(expressio)
```

El valor resultat serà precissament el valor que prengui l'expressió d'aquesta sentència. Un exemple de definició de funció que, donat un natural n, calcula el valor del factorial, $n!$, és

```
factorial <- function(n){
  fac <- 1
  i   <- 1
  while (i <= n){
    fac <- fac * i
    i   <- i + 1
  }
  return(fac)
}
```

2.2 Utilització

Les funcions no s'executen per elles mateixa sinó que cal que hi hagi un programa principal o una altra funció que desencadeni l'execució. La sintaxi del llenguatge **R** per a desencadenar-ne l'execució és molt senzilla: només cal que el nom de la funció aparegui dins d'una expressió de càlcul. En programació de computadors hom acostuma a dir que es fa una *crida* a la funció.

En aquest curs, la crida a una funció cal que inclogui exactament tants paràmetres correctament inicialitzats com hi ha a la definició de la funció. La correspondència de significat entre els paràmetres de la crida i els de la definició serà posicional, és a dir, primer de crida amb primer de definició, segon amb segon i així successivament, com a les funcions a la matemàtica.

El següent exemple il.lustra el càlcul d'un nombre combinatori

$$\binom{a}{b} = \frac{a!}{b!\ (a-b)!}$$

tot utilitzant la funció definida més amunt per a calcular el factorial.

```
a     <- scan(n=1, quiet=TRUE)
b     <- scan(n=1, quiet=TRUE)
afac  <- factorial(a)
bfac  <- factorial(b)
abfac <- factorial(a - b)
cab   <- afac / (bfac * abfac)
cat("cnm : ", cab, "\n")
```

Una manera senzilla, tot i que no és exacta, d'interpretar què és el que passa és la següent. Quan es fa la crida

```
afac <- factorial(a)
```

el valor que tingui la variable a al programa que fa la crida s'assigna al primer paràmetre de la funció, en aquest cas `n`. Tot seguit es desencadena l'execució pròpiament dita. És important recordar que, en aquest curs, les variables d'una funció no tenen cap relació amb les del programa que la crida, tot i que puguin tenir noms coincidents. En acabar l'execució de la funció, el valor que tingui a la funció l'expressió de retorn, en el nostre exemple la variable `fac`, s'assigna a la variable `afac` del programa que fa la crida. La interpretació de la crida `bfac <- factorial(b)` és anàloga.

Una situació un xic més complicada correspon a la crida

```
abfac <- factorial(a - b)
```

la qual requereix una interpretació més elaborada. Abans de desencadenar l'execució de la funció, s'avalua l'expressió `a - b` i el resultat s'assigna al paràmetre `n` de la funció. A partir d'ací, la interpretació coincideix amb la descrita anteriorment. El resultat del càlcul de la funció s'assigna ara a la variable `abfac`.