

COMMAND-LINE ARGUMENTS

Command-line arguments

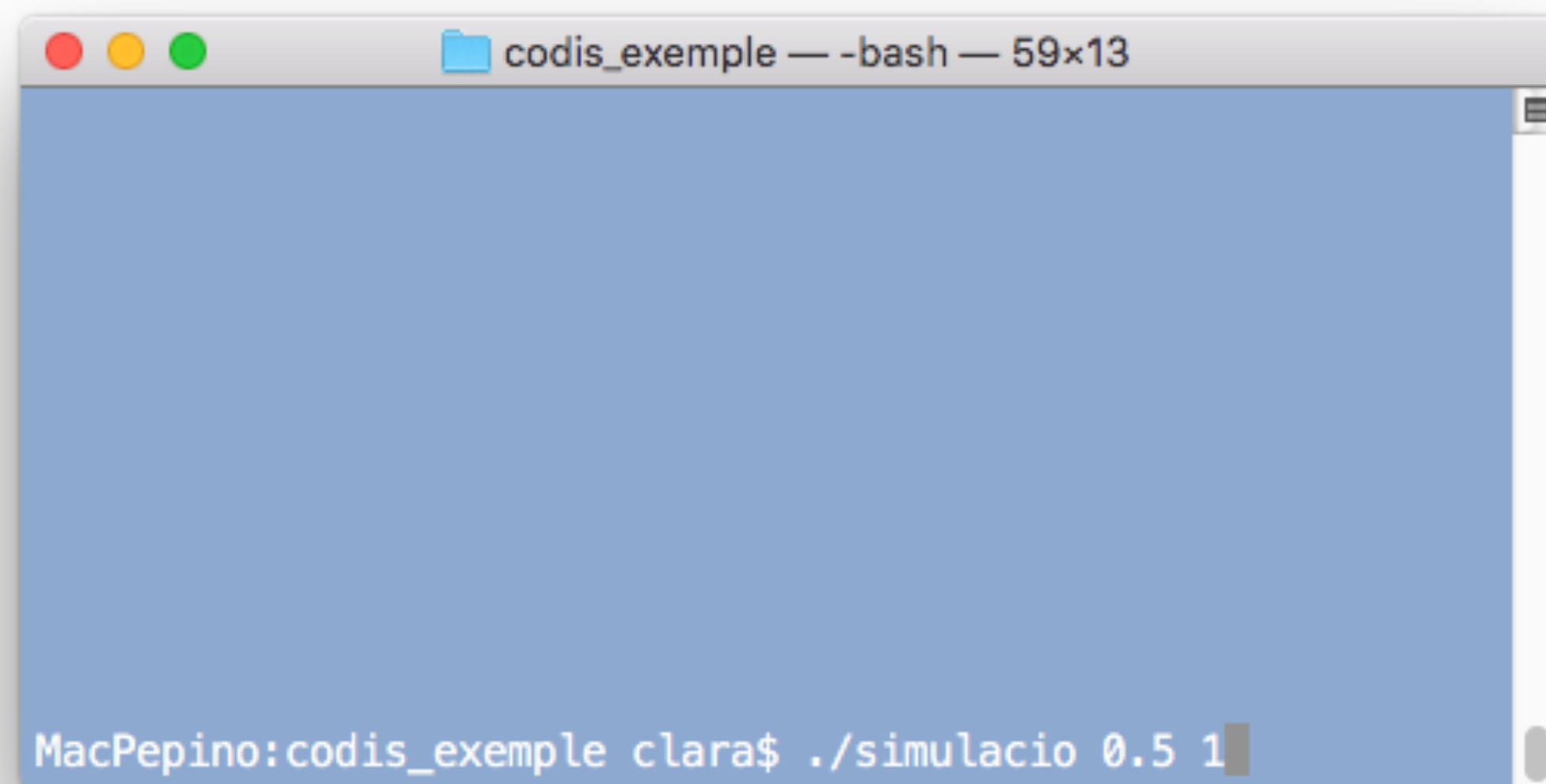
Necessitat

- A vegades serà útil poder passar per línia de comandes alguns paràmetres del nostre programa.
- Maneres de definir valors de paràmetres:
 - Hard-coded (`#define`): poc flexible
 - Preguntar usuari (`scanf`): horrible, pesadíssim per l'usuari, no automatitzable
 - Llegir de fitxer: Tenir un fitxer de configuració que es llegeix sempre. OK.
 - Command-line args: fàcilment automatitzable per bash. OK

Command-line arguments

Necessitat

- A vegades serà útil poder passar per línia de comandes alguns paràmetres del nostre programa.
- Maneres de definir valors de paràmetres:
 - Hard-coded (`#define`): poc flexible
 - Preguntar usuari (`scanf`): horrible, pesadíssim per l'usuari, no automatitzable
 - Llegir de fitxer: Tenir un fitxer de configuració que es llegeix sempre. OK.
 - Command-line args: fàcilment automatitzable per bash. OK



Command-line arguments

Necessitat

- A vegades serà útil poder passar per línia de comandes alguns paràmetres del nostre programa.
- Maneres de definir valors de paràmetres:
 - Hard-coded (`#define`): poc flexible
 - Preguntar usuari (`scanf`): horrible, pesadíssim per l'usuari, no automatitzable
 - Llegir de fitxer: Tenir un fitxer de configuració que es llegeix sempre. OK.
 - Command-line args: fàcilment automatitzable per bash. OK

Què aprendrem?

- Com passar 1 paràmetre per command line
- Com passar diversos paràmetres per command line
- Com fer opcionals els arguments command-line
- Com fer arguments command-line "named"
- Com estructurar la lectura d'arguments command-line en una simulació.

Command-line arguments

NIVELL BÀSIC

Com es fa?



```
#include <stdio.h>
```

```
int main(           ) {
```

```
}
```

Command-line arguments

NIVELL BÀSIC

Com es fa?



```
#include <stdio.h>
```

```
int main( int argc, char *argv[] ) {
```


```
}
```

Command-line arguments

NIVELL BÀSIC

Com es fa?

argc fa referència al nombre
d'arguments rebuts
(argument count)



```
#include <stdio.h>

int main( int argc, char *argv[] ) {
```

```
}
```

Command-line arguments

NIVELL BÀSIC

Com es fa?

argv és la taula on
s'emmagatzemen els
diferents arguments
(argument values)

argc fa referència al nombre
d'arguments rebuts
(argument count)

```
#include <stdio.h>
```

```
int main( int argc, char *argv[] ) {
```

```
}
```


Command-line arguments

NIVELL BÀSIC

Com es fa?

argv és la taula on
s'emmagatzemen els
diferents arguments
(argument values)

argc fa referència al nombre
d'arguments rebuts
(argument count)

```
#include <stdio.h>

int main( int argc, char *argv[] ) {
```

```
}
```

NOTA:

Si canviem els noms "argc" i "argv" per altres noms, també funciona, però no ho hem de fer perquè és una convenció de C.

Command-line arguments

NIVELL BÀSIC

Com es fa?

argv és la taula on
s'emmagatzemen els
diferents arguments
(argument values)

argc fa referència al nombre
d'arguments rebuts
(argument count)

```
#include <stdio.h>

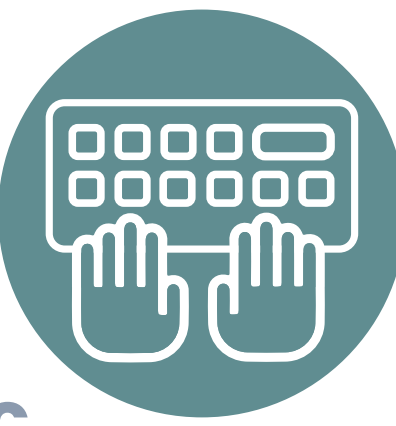
int main( int argc, char *argv[] ) {

    if( argc == 2 ) {
        printf("El primer argument es: %s\n", argv[0]);
        printf("El segon argument es:  %s\n", argv[1]);
    }
    else if( argc > 2 ) {
        printf("Massa arguments!\n");
    }
    else {
        printf("S'espera un argument\n");
    }
}
```

El primer argument sempre
és el nom del programa.
Per tant, encara que no
passem cap argument,
sempre n'hi ha un.

Command-line arguments

NIVELL BÀSIC



genera_sequencia.c

Exemple:

- Farem un programa que generi una seqüència d'enters des de 1..N, on N és l'argument que li passem per la línia de comandes.
- Escriviu el programa **genera_sequencia.c** o baixeu-lo del moodle.
- Comproveu que enteneu les seves parts i com funciona.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main( int argc, char *argv[] ) {
5     int n; // longitud seqüència
6
7     /* Comprovar que l'usuari ens ha passat per paràmetre la longitud*/
8
9     if( argc != 2 ) {
10         printf("Error, la crida es: %s longitud_sequencia\n", argv[0]);
11         return -1;
12     }
13     /* Convertir a enter el paràmetre de longitud */
14     n = atoi(argv[1]);
15
16     /* Generar_sequencia */
17     for(int i=0; i<n; i++){
18         printf("%d ", i+1);
19     }
20     printf("\n");
21     return 0;
22 }
```

Command-line arguments

NIVELL MIG

Com fer arguments opcionals

- Què vol dir arguments opcionals? Que el programa funcionarà igualment encara que l'usuari no introdueixi el valor per paràmetre.
- Això requereix tenir un valor "per defecte" que farem servir si l'usuari no introdueix el seu valor.

Command-line arguments

NIVELL MIG

Com fer arguments opcionals

- Què vol dir arguments opcionals? Que el programa funcionarà igualment encara que l'usuari no introdueixi el valor per paràmetre.
- Això requereix tenir un valor "per defecte" que farem servir si l'usuari no introdueix el seu valor.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define N_DEFAULT 10
4 int main( int argc, char *argv[] ) {
5     int n = N_DEFAULT;
6
7     /* Comprovar que l'usuari ens ha passat per paràmetre la longitud*/
8     if (argc == 2){
9         /* Si tenim 2 arguments, un d'ells és el paràmetre*/
10         n = atoi(argv[1]);
11     }
12     else if(argc > 2){
13         /* Si n'hi ha mes, malament*/
14         printf("Error, la crida es: %s longitud_sequencia\n", argv[0]);
15         return -1;
16     }
17
18     /* Generar_sequencia */
19     for(int i=0; i<n; i++){
20         printf("%d ",i+1);
21     }
22     printf("\n");
23     return 0;
24 }
```

Command-line arguments

NIVELL MIG

Com fer arguments opcionals

- Què vol dir arguments opcionals? Que el programa funcionarà igualment encara que l'usuari no introdueixi el valor per paràmetre.
- Això requereix tenir un valor "per defecte" que farem servir si l'usuari no introdueix el seu valor.
- Modifiqueu el programa anterior perquè l'argument longitud sigui opcional



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define N_DEFAULT 10
4 int main( int argc, char *argv[] ) {
5     int n = N_DEFAULT;
6
7     /* Comprovar que l'usuari ens ha passat per paràmetre la longitud*/
8     if (argc == 2){
9         /* Si tenim 2 arguments, un d'ells és el paràmetre*/
10         n = atoi(argv[1]);
11     }
12     else if(argc > 2){
13         /* Si n'hi ha mes, malament*/
14         printf("Error, la crida es: %s longitud_sequencia\n", argv[0]);
15         return -1;
16     }
17
18     /* Generar_sequencia */
19     for(int i=0; i<n; i++){
20         printf("%d ",i+1);
21     }
22     printf("\n");
23     return 0;
24 }
```


Command-line arguments

NIVELL MIG

Com fer arguments opcionals

- Què vol dir arguments opcionals? Que el programa funcionarà igualment encara que l'usuari no introdueixi el valor per paràmetre.
- Això requereix tenir un valor "per defecte" que farem servir si l'usuari no introdueix el seu valor.
- Modifiqueu el programa anterior perquè l'argument longitud sigui opcional

```
$ ./genera_sequencia 5
1 2 3 4 5

$ ./genera_sequencia 2
1 2

$ ./genera_sequencia
1 2 3 4 5 6 7 8 9 10
```



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define N_DEFAULT 10
4 int main( int argc, char *argv[] ) {
5     int n = N_DEFAULT;
6
7     /* Comprovar que l'usuari ens ha passat per paràmetre la longitud*/
8     if (argc == 2){
9         /* Si tenim 2 arguments, un d'ells és el paràmetre*/
10        n = atoi(argv[1]);
11    }
12    else if(argc > 2){
13        /* Si n'hi ha mes, malament*/
14        printf("Error, la crida es: %s longitud_sequencia\n", argv[0]);
15        return -1;
16    }
17
18    /* Generar_sequencia */
19    for(int i=0; i<n; i++){
20        printf("%d ",i+1);
21    }
22    printf("\n");
23    return 0;
24 }
```

Command-line arguments

NIVELL MIG

Més d'un argument

- Si volem afegir més arguments, podem fer-ho, però sempre respectant l'ordre. L'usuari els haurà d'introduir en el mateix ordre que nosaltres els llegim.
- Per exemple: afegiu un nou paràmetre que sigui l'increment entre un nombre i un altre. Feu que el valor per defecte = 1.

Command-line arguments

NIVELL MIG

Més d'un argument

- Si volem afegir més arguments, podem fer-ho, però sempre respectant l'ordre. L'usuari els haurà d'introduir en el mateix ordre que nosaltres els llegim.
- Per exemple: afegiu un nou paràmetre que sigui l'increment entre un nombre i un altre. Feu que el valor per defecte = 1.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #define N_DEFAULT 10
4 #define INC_DEFAULT 1
5
6 int main( int argc, char *argv[] ) {
7     int j;
8     int n = N_DEFAULT;
9     int inc = INC_DEFAULT;
10    /* Comprovar quins paràmetres ens ha passat */
11    if(argc > 3){
12        printf("Error, la crida es: %s longitud increment\n", argv[0]);
13        return -1;
14    }
15    else if (argc == 3){
16        /* Si tenim 2 arguments, un d'ells és el paràmetre*/
17        n = atoi(argv[1]);
18        inc = atoi(argv[2]);
19    }
20    else if(argc == 2){
21        n = atoi(argv[1]);
22    }
23    /* Generar seqüència */
24    j=1;
25    for(int i=0; i<n; i++){
26        printf("%d ", j);
27        j = j+inc;
28    }
29    printf("\n");
30 }
```

Command-line arguments

NIVELL PRO

Més d'un argument... "named"

- Tenir més d'un paràmetre fa que haguem de respectar sempre l'ordre dels paràmetres.
- Què hauria passat a l'exemple anterior si haguéssim volgut especificar "increment" però deixar "longitud" per defecte? No es pot.
- Seria molt millor poder posar-li noms als arguments, i referir-nos-hi pel seu nom.
- Per exemple:

Command-line arguments

NIVELL PRO

Més d'un argument... "named"

- Tenir més d'un paràmetre fa que haguem de respectar sempre l'ordre dels paràmetres.
- Què hauria passat a l'exemple anterior si haguéssim volgut especificar "increment" però deixar "longitud" per defecte? No es pot.
- Seria molt millor poder posar-li noms als arguments, i referir-nos-hi pel seu nom.
- Per exemple:



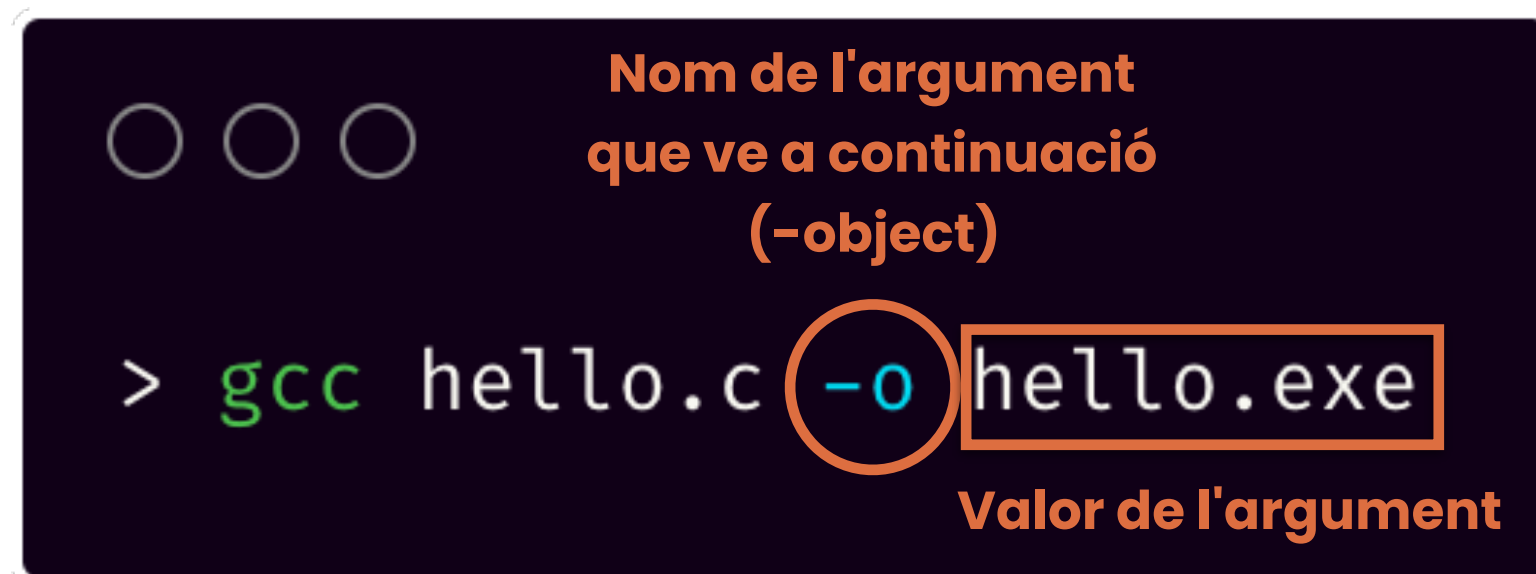
```
> gcc hello.c -o hello.exe
```

Command-line arguments

NIVELL PRO

Més d'un argument... "named"

- Tenir més d'un paràmetre fa que haguem de respectar sempre l'ordre dels paràmetres.
- Què hauria passat a l'exemple anterior si haguéssim volgut especificar "increment" però deixar "longitud" per defecte? No es pot.
- Seria molt millor poder posar-li noms als arguments, i referir-nos-hi pel seu nom.
- Per exemple:



```
> gcc hello.c -o hello.exe
```

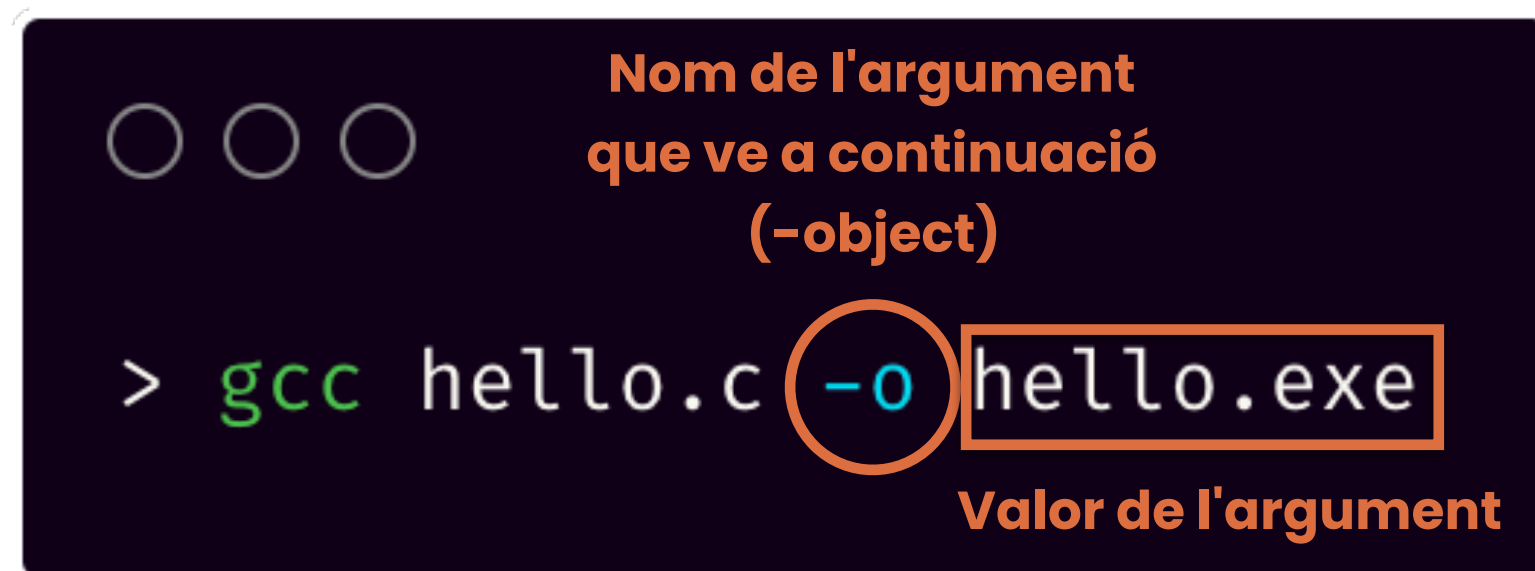
The image shows a terminal window with a dark background. The command `gcc hello.c -o hello.exe` is entered. Above the command, the text "Nom de l'argument que ve a continuació (-object)" is written in orange. Below the command, the text "Valor de l'argument" is written in orange. The flag `-o` is circled in orange, and the value `hello.exe` is boxed in orange.

Command-line arguments

NIVELL PRO

Més d'un argument... "named"

- Tenir més d'un paràmetre fa que haguem de respectar sempre l'ordre dels paràmetres.
- Què hauria passat a l'exemple anterior si haguéssim volgut especificar "increment" però deixar "longitud" per defecte? No es pot.
- Seria molt millor poder posar-li noms als arguments, i referir-nos-hi pel seu nom.
- Per exemple:



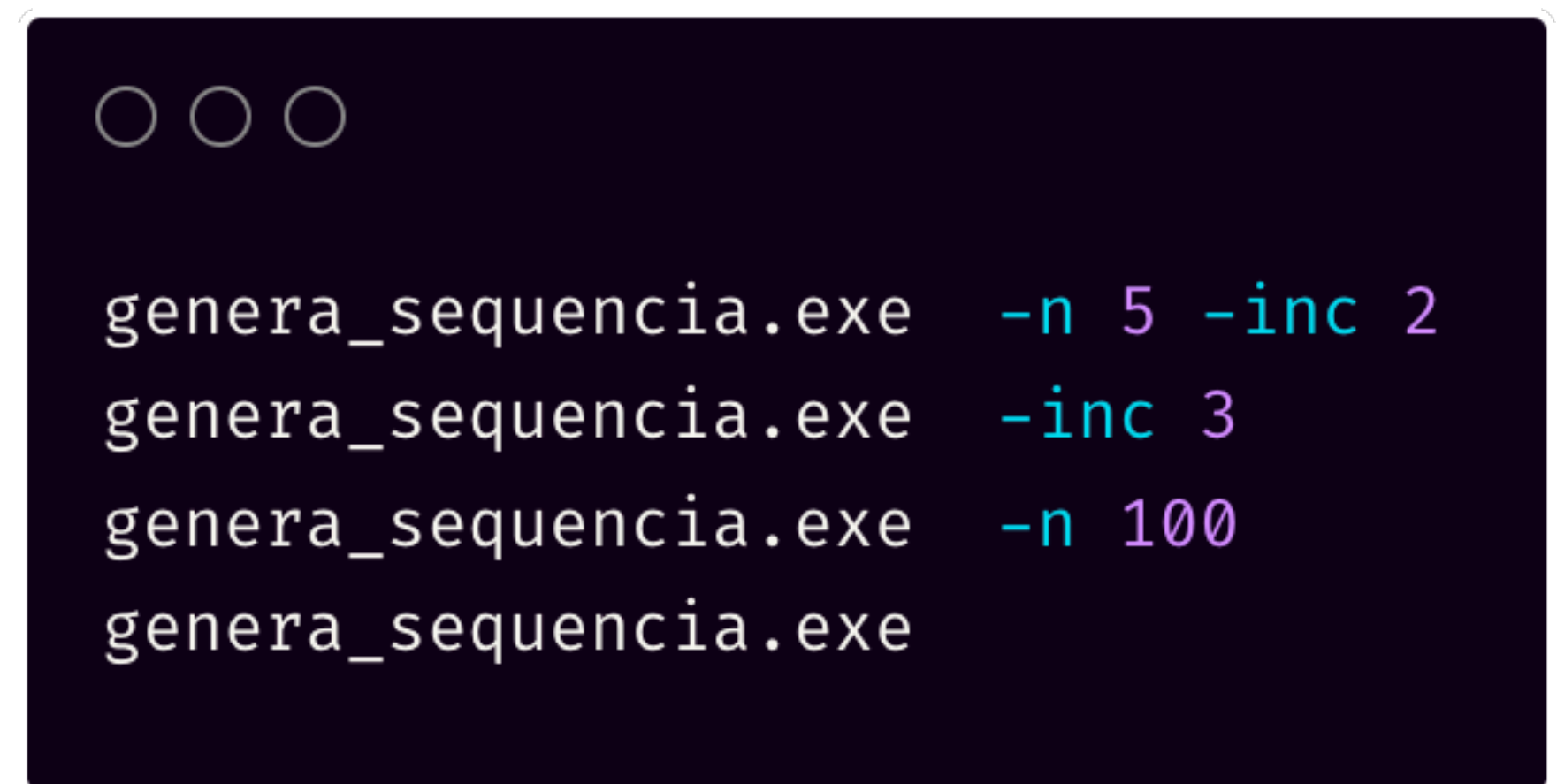
```
> gcc hello.c -o hello.exe
```

Nom de l'argument
que ve a continuació
(-object)

Valor de l'argument

A l'exemple anterior...

- Voldríem aconseguir poder cridar el programa així:



```
genera_sequencia.exe -n 5 -inc 2
genera_sequencia.exe -inc 3
genera_sequencia.exe -n 100
genera_sequencia.exe
```


Command-line arguments

NIVELL PRO

Més d'un argument... "named"

- Necessitarem llegir els arguments per parelles i comprovar els noms dels paràmetres.
- Descarregueu del moolde el programa **genera_sequencia_pro.c**, llegiu-lo, enteneu-lo i executeu-lo.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define N_DEFAULT 10
6 #define INC_DEFAULT 1
7
8
9 int main( int argc, char *argv[] ) {
10     int j;
11     int n = N_DEFAULT;
12     int inc = INC_DEFAULT;
13     int args_tractats;
14
15     /* Comprovar quins paràmetres ens ha passat */
16     // Si argc == 1, agafarem els valors per defecte
17     if (argc > 1){
18         args_tractats = 1; // El nom del programa
19         while(argc - args_tractats > 0){
20             // Llegim primera part de l'argument
21             if (strcmp(argv[args_tractats], "-n")==0){
22                 // El següent argument el guardem a var "n"
23                 n = atoi(argv[args_tractats+1]);
24                 args_tractats += 2;
25             }
26             else if(strcmp(argv[args_tractats], "-inc")==0){
27                 inc = atoi(argv[args_tractats+1]);
28                 args_tractats += 2;
29             }
30             else {
31                 printf("ERROR: No conec aquesta opció\n");
32                 return -1;
33             }
34         }
35     }
36
37     /* Generar seqüència */
38     j=1;
39     for(int i=0; i<n; i++){
40         printf("%d ", j);
41         j = j+inc;
42     }
43     printf("\n");
44     return 0;
45 }
46
```

```

16 // Si argc = 1, agafarem els valors per defecte
17 if (argc > 1){
18     args_tractats = 1; // El nom del programa
19     while(argc - args_tractats > 0){
20         // Llegim primera part de l'argument
21         if (strcmp(argv[args_tractats], "-n")==0){
22             // El següent argument el guardem a var "n"
23             n = atoi(argv[args_tractats+1]);
24             args_tractats += 2;
25         }
26         else if(strcmp(argv[args_tractats], "-inc")==0){
27             inc = atoi(argv[args_tractats+1]);
28             args_tractats += 2;
29         }
30         else {
31             printf("ERROR: No conec aquesta opció\n");
32             return -1;
33         }
34     }
35 }

```

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define N_DEFAULT 10
6 #define INC_DEFAULT 1
7
8
9 int main( int argc, char *argv[] ) {
10     int j;
11     int n = N_DEFAULT;
12     int inc = INC_DEFAULT;
13     int args_tractats;
14
15     /* Comprovar quins paràmetres ens ha passat */
16     // Si argc = 1, agafarem els valors per defecte
17     if (argc > 1){
18         args_tractats = 1; // El nom del programa
19         while(argc - args_tractats > 0){
20             // Llegim primera part de l'argument
21             if (strcmp(argv[args_tractats], "-n")==0){
22                 // El següent argument el guardem a var "n"
23                 n = atoi(argv[args_tractats+1]);
24                 args_tractats += 2;
25             }
26             else if(strcmp(argv[args_tractats], "-inc")==0){
27                 inc = atoi(argv[args_tractats+1]);
28                 args_tractats += 2;
29             }
30             else {
31                 printf("ERROR: No conec aquesta opció\n");
32                 return -1;
33             }
34         }
35     }
36
37     /* Generar seqüència */
38     j=1;
39     for(int i=0; i<n; i++){
40         printf("%d ", j);
41         j = j+inc;
42     }
43     printf("\n");
44     return 0;
45 }
46

```

```
clara$ ./genera_sequencia_pro
```

```
1 2 3 4 5 6 7 8 9 10
```

```
clara$ ./genera_sequencia_pro 100
```

```
ERROR: No conec aquesta opció
```

```
clara$ ./genera_sequencia_pro -n 50
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18  
19 20 21 22 23 24 25 26 27 28 29 30 31 32 33  
34 35 36 37 38 39 40 41 42 43 44 45 46 47 48  
49 50
```

```
clara$ ./genera_sequencia_pro -n 20 -inc 2
```

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33  
35 37 39
```

```
clara$ ./genera_sequencia_pro -inc 3
```

```
1 4 7 10 13 16 19 22 25 28
```

```
clara$ ./genera_sequencia_pro -inc 3 -n 15
```

```
1 4 7 10 13 16 19 22 25 28 31 34 37 40 43
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define N_DEFAULT 10
6 #define INC_DEFAULT 1
7
8
9 int main( int argc, char *argv[] ) {
10     int j;
11     int n = N_DEFAULT;
12     int inc = INC_DEFAULT;
13     int args_tractats;
14
15     /* Comprovar quins paràmetres ens ha passat */
16     // Si argc = 1, agafarem els valors per defecte
17     if (argc > 1){
18         args_tractats = 1; // El nom del programa
19         while(argc - args_tractats > 0){
20             // Llegim primera part de l'argument
21             if (strcmp(argv[args_tractats], "-n")==0){
22                 // El següent argument el guardem a var "n"
23                 n = atoi(argv[args_tractats+1]);
24                 args_tractats += 2;
25             }
26             else if(strcmp(argv[args_tractats], "-inc")==0){
27                 inc = atoi(argv[args_tractats+1]);
28                 args_tractats += 2;
29             }
30             else {
31                 printf("ERROR: No conec aquesta opció\n");
32                 return -1;
33             }
34         }
35     }
36
37     /* Generar seqüència */
38     j=1;
39     for(int i=0; i<n; i++){
40         printf("%d ", j);
41         j = j+inc;
42     }
43     printf("\n");
44     return 0;
45 }
46
```