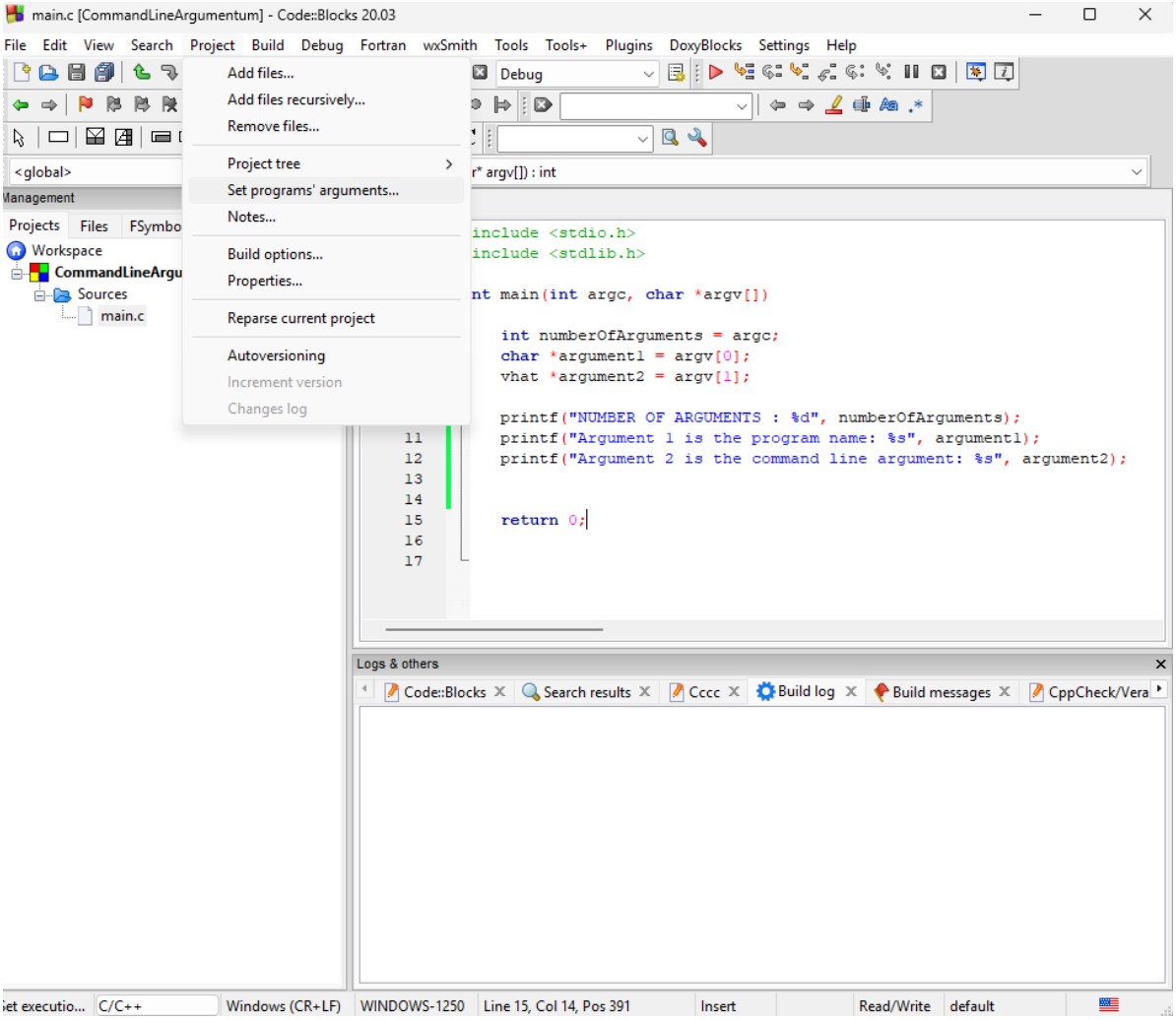
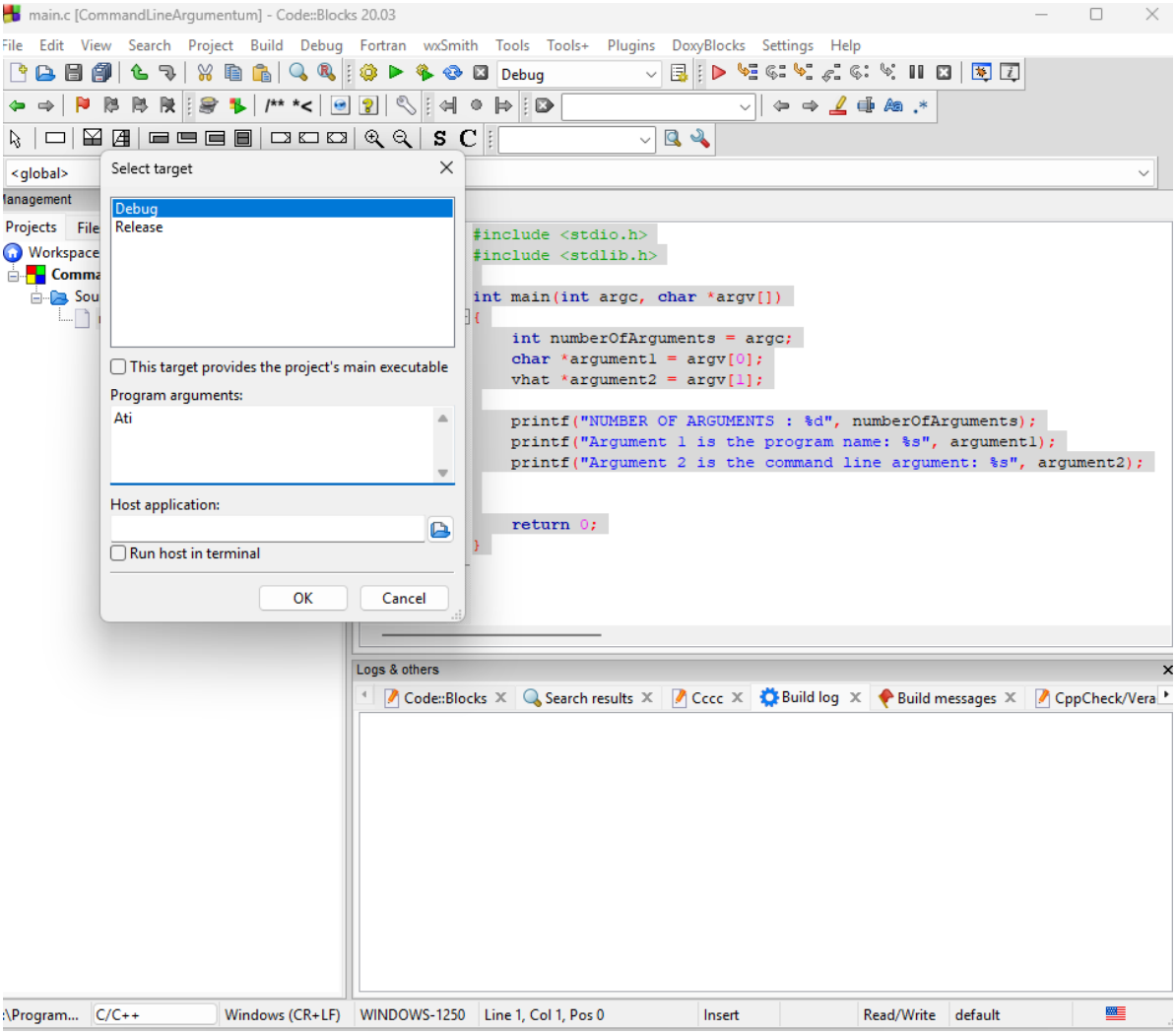
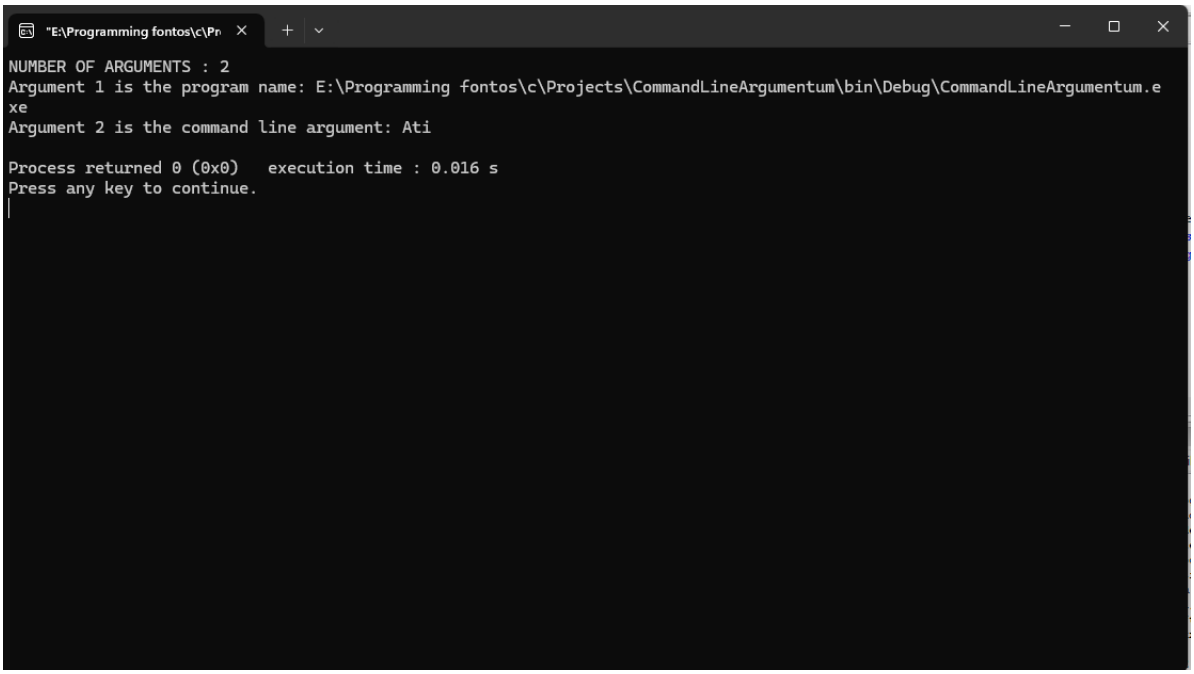


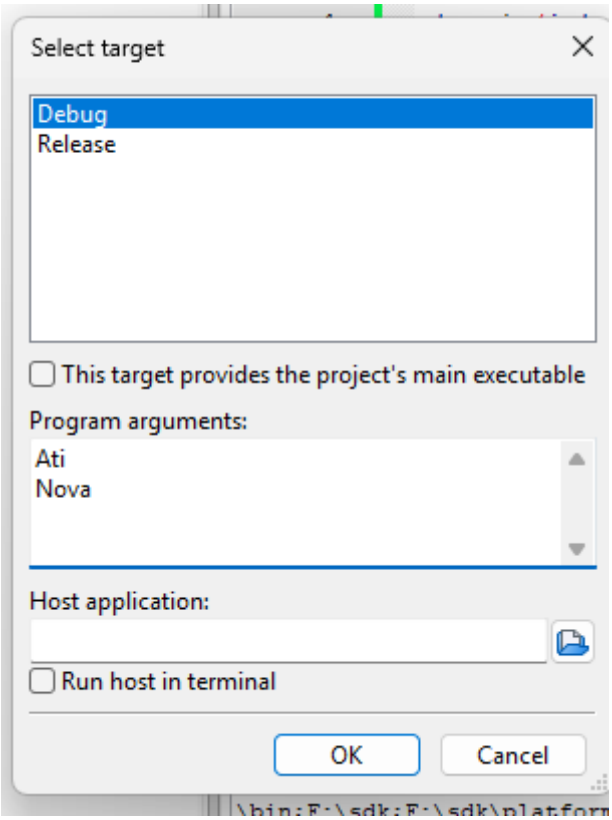
Argumentek átadása:

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

int main(int argc, char *argv[])
{
    int numberOfArguments = argc;
    char *argument1 = argv[0];
    vhat *argument2 = argv[1];

    printf("NUMBER OF ARGUMENTS : %d", numberOfArguments);
    printf("Argument 1 is the program name: %s", argument1);
    printf("Argument 2 is the command line argument: %s", argument2);


    return 0;
}
```



és a program argumentsbe beirjuk mit akarunk átadni milyen adatot.






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

int main(int argc, char *argv[])
{
    int numberOfArguments = argc;
    char *argument1 = argv[0];
    char *argument2 = argv[1];
    char *argument3 = argv[2];

    printf("NUMBER OF ARGUMENTS : %d\n", numberOfArguments);
    printf("Argument 1 is the program name: %s\n", argument1);
    printf("Argument 2 is the command line argument: %s\n", argument2);
    printf("Argument 3 is the command line argument: %s\n", argument3);

    return 0;
}
```

Megjegyzések és esetleges fejlesztések:

1. **Biztonságosabb kód:**
   Jó lenne ellenőrizni, hogy valóban megadtak-e legalább 3 argumentumot (a program neve + 2 argumentum), különben a `argv[1]` vagy `argv[2]` elérése hibát okozhat (memórián kívüli olvasás).

Például így:

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

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

```
{
    printf("NUMBER OF ARGUMENTS : %d\n", argc);

    if (argc > 0)
        printf("Argument 1 is the program name: %s\n", argv[0]);

    if (argc > 1)
        printf("Argument 2 is the command line argument: %s\n", argv[1]);
    else
        printf("Argument 2 is missing!\n");

    if (argc > 2)
        printf("Argument 3 is the command line argument: %s\n", argv[2]);
    else
        printf("Argument 3 is missing!\n");

    return 0;
}
```

1. **Használat:**
   Parancssorban így futtasd:

```
CommandLineArgumentum.exe Ati Nova
```

Ekkor az `argv[1]` az "Ati", az `argv[2]` a "Nova".

Ha csak 0 vagy 1 argumentumot adsz, az előző verzió hibát okozhatott volna.

Ha szeretnéd, tudok segíteni olyan programot írni, ami dinamikusan kezeli az argumentumokat, vagy dolgozik velük.

Jó kérdés, nézzük meg részletesen, mire jó az `argc` és az `argv` a C (és más nyelvek) programjaiban, és hogy miért van erre szükség!

Mi az `argc` és `argv`?

- `argc` (argument count): egy **egész szám**, ami megmutatja, hogy a programot hány argumentummal indították el a parancssorban.
- `argv` (argument vector): egy **karakterláncokból álló tömb** (`char* argv[]`), amely az összes argumentumot tartalmazza szöveges formában. Az első elem (`argv[0]`) a program futtatható állományának neve vagy útvonala, utána jönnek a tényleges paraméterek.

Mire használják ezeket?

1. **Program paraméterezése parancssorból**

   Ezekkel tudod a programodnak megadni, hogy hogyan viselkedjen futás közben. Például:

   - Ha egy fájl nevét vagy elérési útját kell megadni futtatáskor.

- Ha opciókat vagy konfigurációs beállításokat akarsz átadni (pl. `--verbose`, `-o kimenet.txt`).
- Ha dinamikusan szeretnéd szabályozni a program működését, anélkül, hogy újra kellene fordítani.

2. **Példák a gyakorlatból**
   - `grep "keresett_szó" file.txt` → a `grep` program megkapja az `argv`-ban a `"keresett_szó"` és `file.txt` argumentumokat, és az alapján keres.
   - `ls -l /home/user` → az `ls` megkapja a `-l` és `/home/user` argumentumokat.
   - Saját programodnál:
     `myprog input.txt output.txt`
     A program az `argv[1]`-ben az `input.txt`-et, az `argv[2]`-ben az `output.txt`-et fogja látni.
3. **Miért kell ez?**

   Mert így **ugyanaz a program más-más módokon használható anélkül, hogy változtatnod kellene a forráskódon**. Ez egy nagyon fontos és alapvető funkcionalitás minden parancssoros programnál.

Miért külön az `argc`?

- Mert a C nyelvben a `argv` tömb nem tartalmaz semmilyen beépített méret-információt. Csak mutatók egy karakterláncra.
- Az `argc` megmondja, hogy hány ilyen elem van — így lehet biztonságosan végigiterálni a tömbön.

Mire használható a változó?

- Ha például szeretnél parancssorból bemenetet kérni egy fájl nevéhez, vagy egy választáshoz.
- Vagy hogy paraméterezd, mit csináljon a program, pl.:

```c
if (argc > 1 && strcmp(argv[1], "--help") == 0) {
    printf("Használat: myprog [inputfile] [outputfile]\n");
    return 0;
}
```

Összefoglalva:

- **`argc` és `argv` a program belépési pontjánál megadják, hogy hogyan lett a program elindítva.**
- Így a programod dinamikusan tud reagálni különböző bemeneti paraméterekre anélkül, hogy újra kellene fordítani vagy módosítani a kódot.
- Ez alapvető az összes parancssoros programnál.
