

# **RELAZIONE PROGETTO SISTEMI OPERATIVI**

## Informazioni sugli autori

Federico Miniati, 7013975, [federico.miniati@stud.unifi.it](mailto:federico.miniati@stud.unifi.it)

Giacomo De Stefano, 7009345, [giacomo.destefano@stud.unifi.it](mailto:giacomo.destefano@stud.unifi.it)

Matteo Menichetti, 7013974, [matteo.menichetti@stud.unifi.it](mailto:matteo.menichetti@stud.unifi.it)

**Data consegna:** 21/06/2021

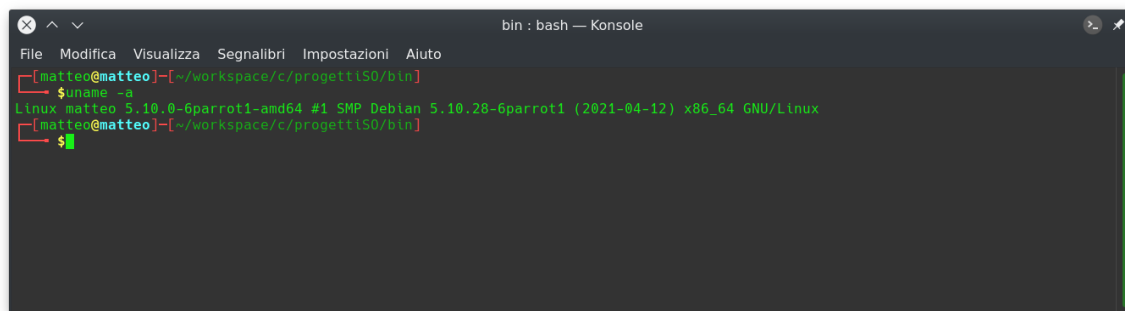
## Sistema obbiettivo

Le caratteristiche hardware dei PC utilizzati per implementare del progetto sono le seguenti:

- CPU i5-4690 RAM 8GB
- CPU R5-3500U RAM 8GB
- CPU i7-11700 RAM 16GB

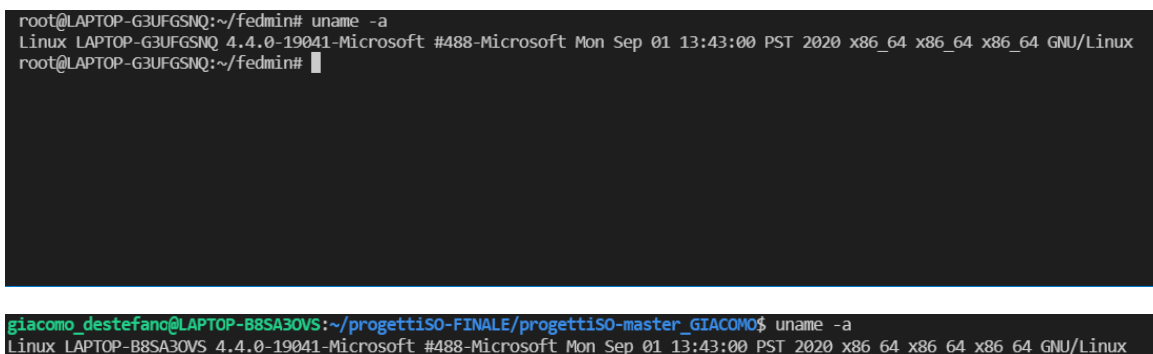
Le caratteristiche software dei PC utilizzati per implementare il progetto sono le seguenti:

Caratteristiche Parrot OS



```
bin : bash — Konsole
File Modifica Visualizza Segnalibri Impostazioni Aiuto
[matteo@matteo]~/workspace/c/progettiSO/bin
$ uname -a
Linux matteo 5.10.0-6parrot1-amd64 #1 SMP Debian 5.10.28-6parrot1 (2021-04-12) x86_64 GNU/Linux
[matteo@matteo]~/workspace/c/progettiSO/bin
$
```

Caratteristiche WSL

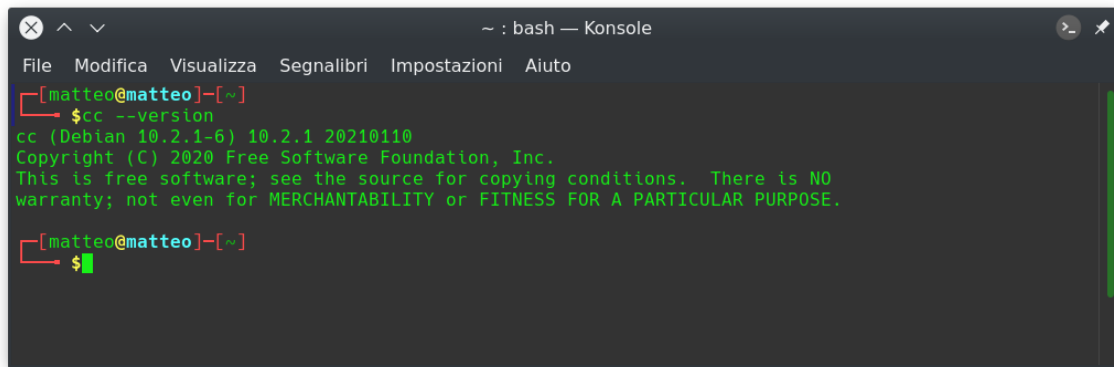


```
root@LAPTOP-G3UFGSNQ:~/fedmin# uname -a
Linux LAPTOP-G3UFGSNQ 4.4.0-19041-Microsoft #488-Microsoft Mon Sep 01 13:43:00 PST 2020 x86_64 x86_64 x86_64 GNU/Linux
root@LAPTOP-G3UFGSNQ:~/fedmin#

giacomo_destefano@LAPTOP-B8SA30VS:~/progettiSO-FINALE/progettiSO-master_GIACOMO$ uname -a
Linux LAPTOP-B8SA30VS 4.4.0-19041-Microsoft #488-Microsoft Mon Sep 01 13:43:00 PST 2020 x86_64 x86_64 x86_64 GNU/Linux
```

È stata utilizzata la funzionalità WSL fornita da Microsoft in Windows per creare sottosistemi Linux. Questo permette di non dover eseguire una macchina virtuale per utilizzare tutto ciò che è necessario per l'implementazione del progetto.

La versione del compilatore utilizzato è la seguente:



```
~ : bash — Konsole
File Modifica Visualizza Segnalibri Impostazioni Aiuto
[matteo@matteo]~$ gcc --version
cc (Debian 10.2.1-6) 10.2.1 20210110
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
[matteo@matteo]~$
```

## Elementi facoltativi

Elemento Facoltativo	Realizzato	Metodo o file principale
Invio di I_AM_ALIVE e realizzazione watchdog.c	Si	watchdog.c
Failure Manager comanda il riavvio di P1, P2, P3	NO	

## Istruzioni per la compilazione

Per la compilazione è previsto l'utilizzo del makefile ed è NECESSARIO eseguire i seguenti passi sul proprio terminale:

- 1) Entrare nella cartella PROGETTO tramite "cd path/PROGETTO" ;
- 2) Eseguire "make" ;
- 3) Eseguire "make install" .

path è da sostituire con il percorso della cartella relativa o assoluta PROGETTO.

Tramite make vengono compilati gli eseguibili corrispondenti ai componenti definiti dal testo del progetto ed creati i file oggetto necessari per la compilazione degli eseguibili.

Tramite “make install” i file generati durante la fase di compilazione saranno spostati nelle cartelle di loro competenza tramite la seguente suddivisione:

- bin conterrà i file eseguibili;
- src conterrà i file sorgenti;
- lib conterrà gli header file utili per i sorgenti;
- tmp conterrà i file oggetto e i meccanismi di comunicazione tra processi necessari per l'esecuzione;
- logs conterrà i file di log.

### **Istruzioni per l'esecuzione**

Il comando NECESSARIO per l'esecuzione corretta del progetto è il seguente: `./main MODALITÀ path/dataset.csv`.

MODALITÀ del comando deve essere sostituita con “FALLIMENTO” o “NORMALE” e path è da sostituire con il percorso di dataset.csv.

Avviare il progetto in bin è NECESSARIO in quanto sono stati utilizzati indirizzi relativi per la gestione dei file di log ed i meccanismi di comunicazione tra processi.

Quando è terminata l'esecuzione del progetto è possibile eliminare tutti i file correlati all'esecuzione (le cartelle bin, tmp e logs) tramite il comando “make clean”. Il comando è NECESSARIO che sia eseguito alla radice della cartella PROGETTO.

### **Progettazione ed implementazione**

La soluzione presentata prevede, per ogni componente, un eseguibile con le funzionalità necessarie per la corretta esecuzione ovvero per ogni componente sarà avviato un corrispettivo processo. Ad esempio: Input Manager è rappresentato dall'eseguibile `input_manager`, P1, P2 e P3 da `p` e così via per Watchdog, Failure Manager e Decision Function.

Il progetto viene avviato tramite l'eseguibile `main`, il quale ha come unico scopo quello di avviare i seguenti processi: `p`, `decision_function` ed `input_manager`.

Per semplificare la struttura dei processi P questi vengono compilati come unico eseguibile ed in base ai parametri ricevuti (0, 1 o 2) dal processo `main` saranno eseguite le funzionalità del corrispettivo P. Le funzionalità richieste sono state divise per processo come segue:

- p1.c definisce le funzionalità del processo P1;
- p2.c definisce le funzionalità del processo P2;
- p3.c definisce le funzionalità del processo P3;
- p.c definisce il comportamento che deve essere tenuto (esecuzione di p1, p2 o p3) e implementa le funzionalità di somma dei caratteri con o senza errore comuni ai p.

I quattro sorgenti saranno compilati come unico eseguibile con il nome di "p".

Inoltre è stato utilizzato l'header file p.h per definire i prototipi, header file e macro utilizzate in p.c, p1.c, p2.c e p3.c

La comunicazione verso il Decision Function dei valori interi, derivati dalla somma della stringa ricevuta da Input Manager, è effettuata tramite pipe con nome.

Le funzionalità di Decision Function sono implementate in decision\_funtion.c il quale verrà compilato come decision\_function.

Il componente avvia Watchdog e Failure Manager in quanto sono strettamente legati a Decision Function poiché comunica con Failure Manager e Watchdog tramite segnali, rispettivamente, il risultato del voto di maggioranza ed il messaggio I\_AM\_ALIVE.

Le funzionalita di Failure Manager sono implementate in failure\_manager.c il quale verrà compilato come failure\_manager.

Il componente attende solamente che gli sia recapitato il segnale SIGUSR1 per terminare ogni processo appartenente al proprio gruppo.

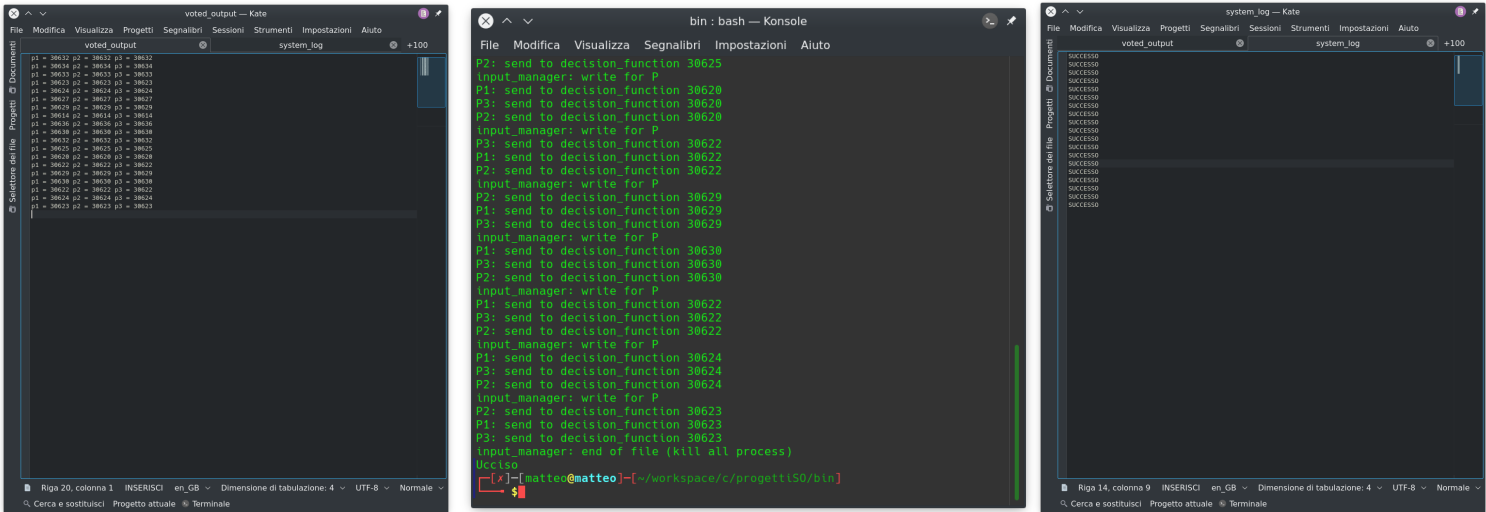
## Esecuzione

### Esecuzione con flag NORMALE

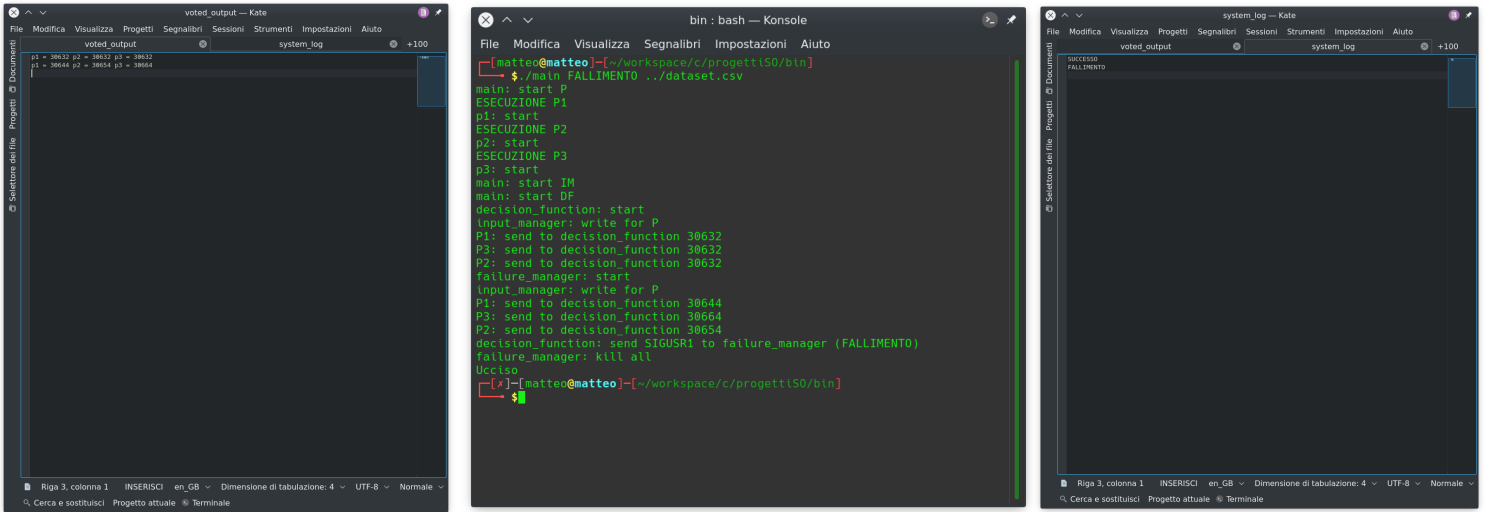
```

[matteo@matteo]~[./workspace/c/progetti/50/bin]
$ ./main NORMALE ../dataset.csv
main: start P
ESECUZIONE P1
p1: start
ESECUZIONE P2
p2: start
ESECUZIONE P3
p3: start
main: start IM
main: start DF
decision_function: start
input_manager: write for P
P1: send to decision_function 30632
P3: send to decision_function 30632
P2: send to decision_function 30632
failure_manager: start
  
```

Questa esecuzione è stata avviata con la modalità NORMALE ed il file dataset.csv composto dalle prime 20 righe del dataset.csv fornito. Il dataset ridotto sarà comune anche alle prossime esecuzioni. Come è possibile notare decision\_function scrive correttamente sui file di log il risultato delle somme. Nelle successive immagini è possibile notare la corretta esecuzione dei processi e la scrittura nei file di log oltre alla terminazione di tutti i processi da parte del processo input\_manager, quando arriva a fine del file attraverso la sua lettura.



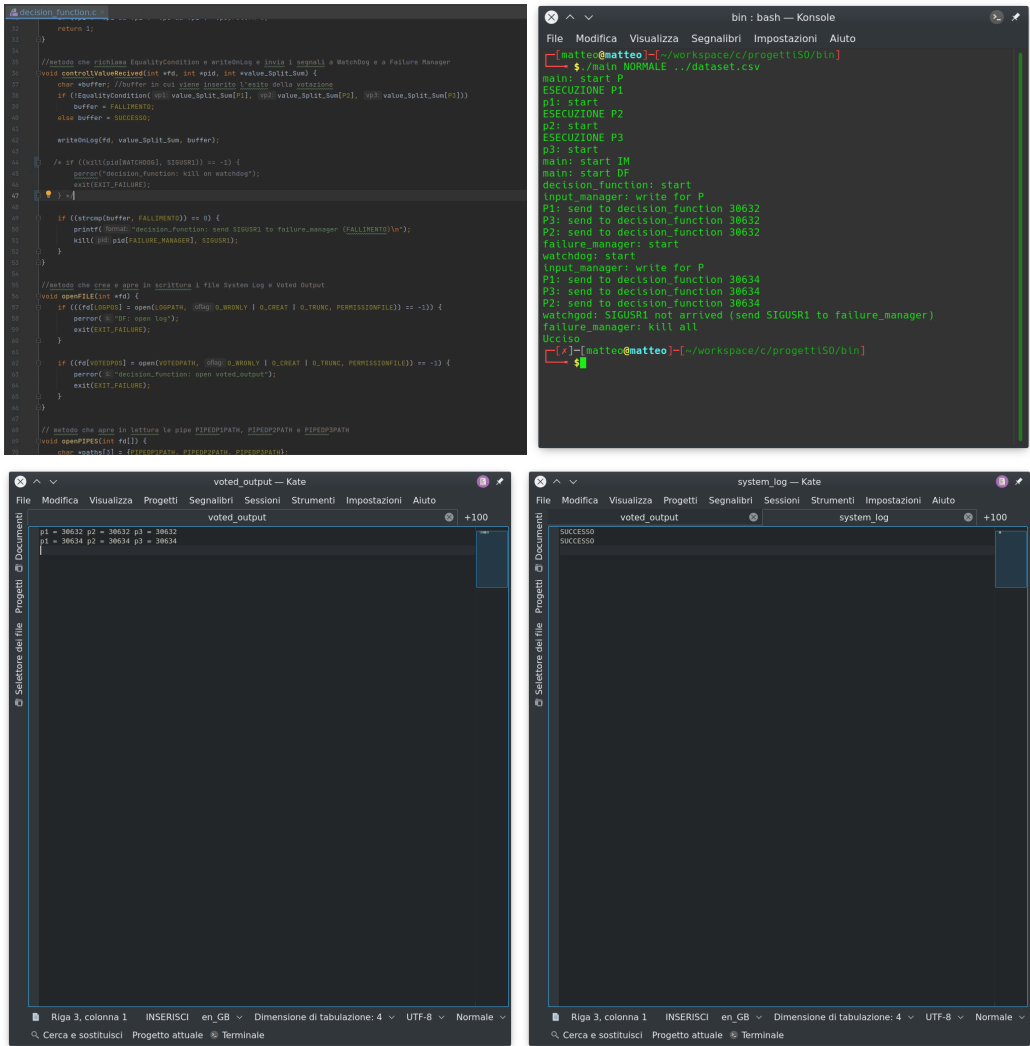
### Esecuzione con flag FALLIMENTO



Questa esecuzione è stata avviata con la modalità FALLIMENTO. Com'è possibile notare quando il processo decision\_function riceve 3 valori discordi invia il segnale SIGUSR1 al processo failure\_manager e scrive sui file di log. failure\_manager termina tutti i processi.

## Esecuzione Watchdog

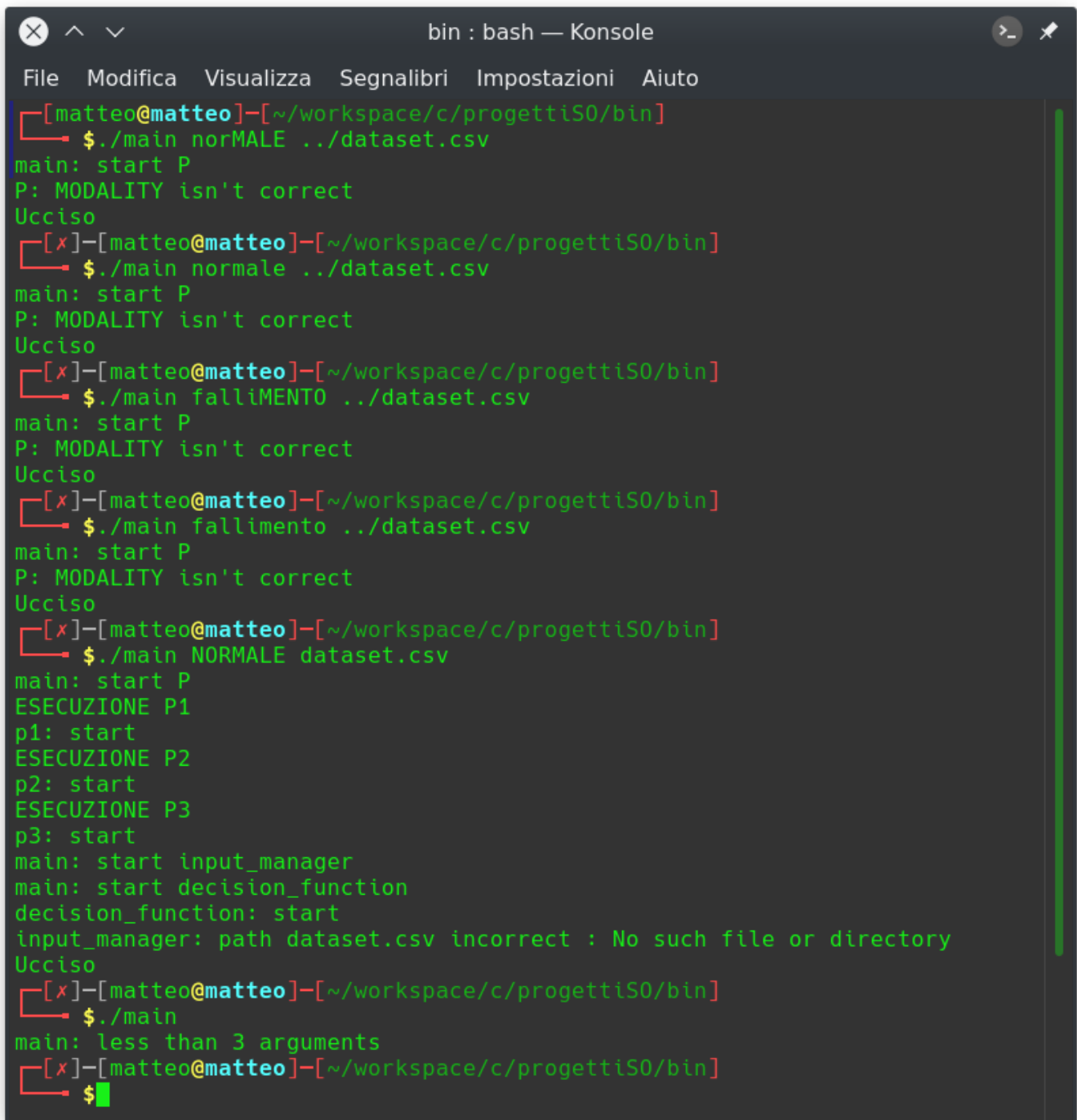
Per mostrare la corretta esecuzione del watchdog è stato rimossa l'istruzione utilizzata per inviare il segnale SIGUSR1 (I\_AM\_ALIVE) al watchdog e dopo due periodi senza ricevere il segnale invia il segnale SIGUSR1 a failure\_manager.



## Esecuzione con parametri sbagliati

In seguito mostriamo l'esecuzione con i parametri non corretti:

- 1) Invece di NORMALE noRMALE e normale;
- 2) Invece di FALLIMENTO falliMENTO e fallimento;
- 3) Esecuzione con dataset.csv sbagliato;
- 4) Esecuzione senza parametri.



```
bin : bash — Konsole
File Modifica Visualizza Segnalibri Impostazioni Aiuto
[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$ ./main norMALE ../dataset.csv
main: start P
P: MODALITY isn't correct
Ucciso
[x]-[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$ ./main normale ../dataset.csv
main: start P
P: MODALITY isn't correct
Ucciso
[x]-[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$ ./main falliMENTO ../dataset.csv
main: start P
P: MODALITY isn't correct
Ucciso
[x]-[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$ ./main fallimento ../dataset.csv
main: start P
P: MODALITY isn't correct
Ucciso
[x]-[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$ ./main NORMALE dataset.csv
main: start P
ESECUZIONE P1
p1: start
ESECUZIONE P2
p2: start
ESECUZIONE P3
p3: start
main: start input_manager
main: start decision_function
decision_function: start
input_manager: path dataset.csv incorrect : No such file or directory
Ucciso
[x]-[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$ ./main
main: less than 3 arguments
[x]-[matteo@matteo]~[~/workspace/c/progettiSO/bin]
$
```