Corso di Laboratorio di Programmazione
Progetto finale – Sistema domotico

Per questo progetto vi viene richiesto di sviluppare un programma in C++ che implementa un sistema di gestione e monitoraggio del consumo energetico di dispositivi domotici. Il sistema dovrà mappare i consumi di vari elettrodomestici allacciati alla rete e verificare che non sia superato il valore soglia di assorbimento massimo, eventualmente intervenendo con le modalità che saranno spiegate sotto.

1. Dispositivi domotici

I dispositivi domotici che il sistema deve gestire sono elencati nella tabella sottostante. I dispositivi sono di due categorie:

- CP (Ciclo Prefissato): dopo aver ricevuto un input dall'utente, eseguono un ciclo di durata prefissata ("Durata accensione"), durante il quale hanno una richiesta di potenza costante, e poi si spengono; è anche possibile uno spegnimento manuale anticipato;
- Manuali (M): hanno un funzionamento manuale, pertanto si accendono e si spengono in risposta a input dell'utente; finché sono in funzione assorbono una potenza costante.

Tutti i dispositivi devono avere la possibilità di definire un orario per l'accensione automatica posticipata e, per la categoria M, un orario per lo spegnimento (per i CP vale la durata del ciclo). Esiste anche un impianto fotovoltaico che inserisce potenza costante nel sistema, sempre in base all'input dell'utente. Il limite massimo di potenza che si può assorbire dalla rete è fornito in fase di inizializzazione al sistema (tramite una costante definita nel codice o un argomento da riga di comando) ed è pari a 3,5kW. L'assorbimento di potenza massimo deve essere in ogni momento inferiore alla somma di tale limite con la produzione fotovoltaica.

Dispositivo domotico	**Durata accensione**	**Produzione/Consumo di potenza**
Impianto fotovoltaico	Manuale	+1,5 kW
Lavatrice	1 ora e 50 minuti	- 2 kW
Lavastoviglie	3 ore e 15 minuti	- 1,5 kW
Pompa di calore + termostato	Manuale	- 2 kW
Tapparelle elettriche	1 minuto	- 0,3 kW
Scaldabagno	Manuale	- 1 kW
Frigorifero	Manuale	- 0,4 kW
Forno a microonde	2 minuti	- 0,8 kW
Asciugatrice	1 ora	- 0,5 kW
Televisore	1 ora	- 0,2 kW

Esempio:
All'avvio, il sistema permette un consumo massimo di potenza di 3,5 kW grazie all'allaccio alla rete elettrica. Consideriamo che il frigorifero sia sempre acceso (-0.4 kW). Alle 07:00 di mattina vengono accesi la pompa di calore (-2 kW) e lo scaldabagno (-1 kW). A questo punto, il consumo di potenza del sistema è pari a 3,4 kW e l'eventuale accensione di un altro dispositivo (ad esempio il microonde) comporterebbe un assorbimento oltre la soglia massima di 3,5kW e innescherebbe la policy di spegnimento dei dispositivi. Tuttavia alle 08:00 viene attivato l'impianto fotovoltaico che incrementa il valore del consumo massimo a 5kW. In questo scenario, risulterebbe possibile, ad esempio, l'ulteriore accensione in contemporanea del microonde (-0,8 kW), delle tapparella (-0,3 kW) oppure l'accensione della lavastoviglie (-1,5 kW).

Dispositivi:
Tutti i dispositivi possiedono le seguenti informazioni: nome, ID, consumo o produzione energetica. L'ID di ogni dispositivo deve essere univoco. Per ottimizzare la gestione dei dispositivi ed evitare la duplicazione del codice, è importante organizzare le classi che rappresentano i dispositivi in una gerarchia, **utilizzando l'ereditarietà e le funzioni virtuali**.
I dispositivi riportati nella tabella precedente sono quelli richiesti dal progetto. In caso è possibile prevedere dispositivi aggiuntivi (le loro caratteristiche devono essere chiaramente riportate nel README) o aggiungere più dispositivi di uno stesso tipo (ad esempio 2 televisori). In questo caso, dispositivi dello stesso tipo devono avere anche un nome diverso (ad esempio Televisore1 e Televisore2).

Da potenza a consumo energetico:
I valori riportati nella tabella precedente fanno riferimento al consumo o alla produzione di potenza dei dispositivi. Per il calcolo della produzione o del consumo energetico basta moltiplicare tale valore per il numero di ore (o frazioni di ora) in cui il dispositivo è rimasto acceso. Ad esempio, una lavatrice che necessita di potenza di 2 kW, se rimane accesa per 3 ore e 15 minuti comporterà un consumo energetico di (3 ore * 2 kW) + 0.25 ore * 2 kW = 6.5 kWh.

2. Funzionamento
Gli input dall'utente citati sopra sono espressi secondo le modalità seguenti.

2.1 Interfaccia utente
Dopo l'avvio e le relative procedure di setup, il sistema rimane in attesa di input dall'utente tramite terminale (su standard input). Sono previsti i seguenti comandi testuali:

Comando	Descrizione
`set ${DEVICENAME} on` **DomoticDevice::turnOn()**	Accende il dispositivo (sia CP che M).
`set ${DEVICENAME} off` **DomoticDevice::turnOff()**	Spegne il dispositivo.
`set ${DEVICENAME} ${START} [${STOP}]` **DomoticDevice::setTimer()** **FixedCycleDevice::setTimer()**	Imposta l'orario di accensione e spegnimento per il dispositivo. L'orario di spegnimento è disponibile per solo i dispositivi M.

`rm ${DEVICENAME}`	**FixedCycleDevice::stopCycle()**	Rimuove il timer associato al dispositivo.
`show`	**DomoticSystem::operator<<**	Mostra la lista di tutti i dispositivi (attivi e inattivi) con la produzione/consumo energetico di ciascuno dalle 00:00 al momento di invio del comando. Inoltre mostra la produzione/consumo energetico totale del sistema dalle 00:00 al momento di invio del comando
`show ${DEVICENAME}`	**DomoticDevice::operator<<**	Mostra a schermo produzione/consumo energetico di uno specifico dispositivo.
`set time ${TIME}`	**Time::setTime()**	Va a una specifica ora del giorno (vedi sotto).
`reset time`	**Time::resetTime()**	**Comando per il debug**. Resetta il tempo del sistema, riportandolo all'orario 00:00. Riporta tutti i dispositivi alle condizioni iniziali. Gli eventuali timer aggiunti dopo l'avvio del sistema vengono mantenuti.
`reset timers`	**DomoticSystem::resetTimers()**	**Comando per il debug**. Rimuove i timer di tutti i dispositivi. Tutti i dispositivi rimangono nel loro stato attuale (accesi o spenti).
`reset all`	**DomoticSystem::resetAll()**	**Comando per il debug**. Riporta il sistema alle condizioni iniziali. L'orario viene impostato a 00:00, tutti i timer vengono rimossi. Tutti i dispositivi vengono spenti.

${DEVICENAME}: Stringa di testo indicante il nome del device
${START}*, *${STOP}, *${TIME}*: orario della giornata hh:mm (formato 24 ore)

Il programma dovrà verificare la correttezza sintattica e logica del comando (case sensitive).

2.2 Gestione del tempo

Il trascorrere del tempo viene gestito in risposta al comando "`set time ${TIME}`". Ad ogni utilizzo di questo comando, il sistema deve simulare il trascorrere del tempo a passi di 1 minuto. All'avvio del programma l'orario corrente è 00:00. Con il comando "`set time`" è possibile solamente fornire un orario nel futuro (dovrà quindi essere rigettato il caso in cui venga fornito un orario anteriore rispetto all'orario attuale del sistema). Si consideri solamente l'arco temporale di una giornata (dalle 00:00 alle 23:59) alla fine della quale il programma terminerà. I dispositivi CP che prevedano un orario di spegnimento oltre le 23:59 rimarranno accesi fino alla fine di esecuzione del programma. Nel caso in cui nell'arco temporale definito dal comando "`set time`" il timer di un dispositivo si attivi o disattivi, il programma deve riportare all'utente (stampando a schermo) l'evento.

Ad esempio, ipotizziamo che il sistema sia composto da due dispositivi, un termostato con timer di accensione fra le 07:00 e le 07:30 e una lavatrice con partenza programmata alle 18:00 come illustrato di seguito:

Nome dispositivo	**Accensione**	**Spegnimento**	**Consumo**
Pompa di calore con termostato	07:00	07:30	- 2 kW
Lavatrice	18:00	-	- 2 kW

Nel caso l'utente invii il comando:

```
>> set time 08:50
```

l'output dal sistema dovrà risultare (si veda la sezione 3 per ulteriori dettagli sul formato di logging):

```
[00:00] L'orario attuale è 00:00
[07:00] Il dispositivo "Pompa di calore con termostato" si è acceso
[07:30] Il dispositivo "Pompa di calore con termostato" si è spento
[08:50] L'orario attuale è 08:50
```

Nel caso l'utente invii un ulteriore comando:

```
>> set time 19:15
```

l'output sistema dovrà risultare:

```
[08:50] L'orario attuale è 08:50
[18:00] Il dispositivo 'Lavatrice' si è acceso
[19:15] L'orario attuale è 19:15
```

Un ulteriore comando:

```
>> set time 20:45
```

porterà al seguente output:

```
[19:15] L'orario attuale è 19:15
[19:50] Il dispositivo 'Lavatrice' si è spento
[20:45] L'orario attuale è 20:45
```

2.3 Casi particolari

È possibile che si verifichino dei casi particolari durante l'esecuzione del programma. Le politiche di gestione di questi casi vengono riportati nella tabella di seguito:

Descrizione	**Politica**
L'utente spegne manualmente un dispositivo (tramite il comando "`set ${DEVICENAME} off`") in un intervallo di tempo in cui era acceso.	Il dispositivo si spegne (riportando a schermo l'azione eseguita) e l'eventuale timer di spegnimento viene ignorato.
In un qualsiasi momento, la potenza assorbita da tutti i dispositivi è maggiore della potenza massima che si può assorbire dalla rete sommata alla potenza aggiuntiva fornita dall'impianto fotovoltaico.	Il sistema inizia a spegnere i dispositivi nell'ordine inverso rispetto all'accensione fino ad ottenere una potenza assorbita minore di quella prodotta, riportando a schermo le azioni eseguite. È possibile anche modificare questa politica con alcune eccezioni (per esempio, evitare di spegnere il frigorifero).
Viene rimosso il timer da un dispositivo mentre è in funzione	Il dispositivo rimane in funzione

3. Eventi e Logging

Ogni evento che accade durante l'esecuzione (ad esempio, i comandi dell'utente, l'accensione e lo spegnimento automatico) deve essere opportunamente stampato a schermo, seguendo il seguente formato: [ORARIOSISTEMA] DESCRIZIONE_EVENTO

Nella tabella di seguito viene riportata una lista di eventi stampati a schermo (viene riportata un orario fittizio, 13:00):

Evento	**Messaggio**
All'esecuzione di ogni comando	[13:00] L'orario attuale è 13:00
Comando accensione [o spegnimento] di un dispositivo	[13:00] Il dispositivo '*${DEVICENAME}*' si è acceso [o spento]
Accensione o spegnimento automatico di un dispositivo	[13:00] Il dispositivo '*${DEVICENAME}*' si è acceso [o spento]
Comando per andare a un determinato orario	[13:00] L'orario attuale è 13:00 (alla fine dell'esecuzione)
Comando per resettare l'orario del sistema	[00:00] L'orario attuale è 00:00
Comando per definire un timer di accensione e spegnimento per un dispositivo	[13:00] Impostato un timer per il dispositivo '*${DEVICENAME}*' dalle XX:YY alle ZZ:WW
Comando per rimuovere il timer da un dispositivo	[13:00] Rimosso il timer dal dispositivo '*${DEVICENAME}*'

Comando per mostrare il consumo attuale di un dispositivo	[13:00] Il dispositivo '*${DEVICENAME}'* ha attualmente consumato XX kWh
Comando per mostrare il consumo attuale di tutti i dispositivi	[13:00] Attualmente il sistema ha prodotto XX kWh e consumato YY kWh. Nello specifico: - Il dispositivo '${DEVICENAME}' ha consumato XX kWh - Il dispositivo '*${DEVICENAME}'* ha consumato XX kWh - … - …

Qualsiasi messaggio stampato a schermo durante l'esecuzione dovrà essere anche salvato su un file di testo di log. Rispetto alla lista fornita, è possibile aggiungere ulteriori eventi di log (se strettamente necessario).

4. Note allo sviluppo

Il progetto sviluppato **deve** essere gestito tramite CMake e compilabile sulla macchina virtuale (VLab/Taliercio.2020). Non è necessario sviluppare tutto sulla macchina virtuale, ma è importante fare verifiche periodiche per essere sicuri di non aver utilizzato codice fuori standard.

4.1 Indicazioni per lo svolgimento

Il progetto deve essere suddiviso in più file sorgente. **Ogni file deve essere scritto da un solo membro del gruppo**. È tuttavia possibile controllare che il codice dei propri compagni di gruppo funzioni correttamente. Un errore in un file che inficia il funzionamento del progetto potrebbe causare una penalizzazione della valutazione dell'intero gruppo. **Il nome dell'autore deve essere indicato in un commento a inizio file**. Ovviamente, è necessario e positivo che si discuta all'interno di ciascun gruppo su come realizzare il codice, ma ogni membro è responsabile della gestione del codice che deve scrivere.

Saranno valutati:

- Chiarezza e correttezza del codice;
- Corretta gestione della memoria e delle strutture dati;
- Efficienza del codice e della soluzione trovata;
- Utilizzo di strumenti appropriati.

Il codice deve essere adeguatamente commentato.

Il software deve essere basato unicamente sulla libreria standard del C++.

4.2 Plagi

Il codice verrà controllato per verificare eventuali plagi tra gruppi appartenenti allo stesso canale e in canali diversi. Verrà effettuato anche un controllo rispetto a codice disponibile online.

4.3 Consegna

Il compito deve essere consegnato su moodle (consegna di gruppo come per la prova intermedia).

Il progetto dovrà essere compresso in un archivio che include una sola directory contenente:

- Il codice sorgente (eventualmente organizzato in sottodirectory);
- Il file CMakeLists.txt necessario alla compilazione (uno solo e posto nella directory principale del progetto);
- Due file di log di esempio;
- Un file readme.txt in cui sono riportate eventuali note che si desidera comunicare in fase di correzione. Questo file non è la documentazione, che deve essere inserita sotto forma di commenti nel codice, ma un elenco di note aggiuntive per chi corregge, per esempio problemi riscontrati e non risolti.

L'archivio non deve contenere l'eseguibile, perché il sorgente sarà compilato in fase di correzione. Il sistema CMake deve compilare con le opzioni di ottimizzazione attivate (-O2).
Dopo la consegna su moodle, **è necessario verificare ciò che è stato consegnato** con i passi seguenti:

- Scaricare il progetto da moodle in una directory diversa da quella usata per sviluppare;
- Lanciare CMake;
- Compilare il codice e verificarne la corretta esecuzione.

Si suggerisce di consegnare anche compiti non completi. È tuttavia necessario che il software consegnato sia **compilabile** ed **eseguibile**.

La consegna in ritardo degli elaborati sarà **fortemente penalizzata** e considerata solo in un brevissimo lasso temporale dopo la scadenza (pochi minuti). Consegne effettuate successivamente saranno ignorate.

Alcuni esempi (non esaustivi) delle voci nella griglia di valutazione del progetto:

- Il progetto non compila
- Il progetto non esegue (dà sempre segmentation fault)
- Il progetto produce segmentation fault occasionali
- Il progetto ha memory leak
- CMakeLists sbagliato o inesistente
- Erroneo (assente o eccessivo) uso dell'ereditarietà
- Erroneo uso della std library per gestire elementi multipli
- Mancato controllo bound array / accesso a memoria non consentito
- Mancato controllo argomenti da riga di comando
- Utilizzo allocazione dinamica in maniera non opportuna
- Il programma non funziona correttamente
- Bad code style lieve/grave
- Mancanza log file richiesto dal progetto