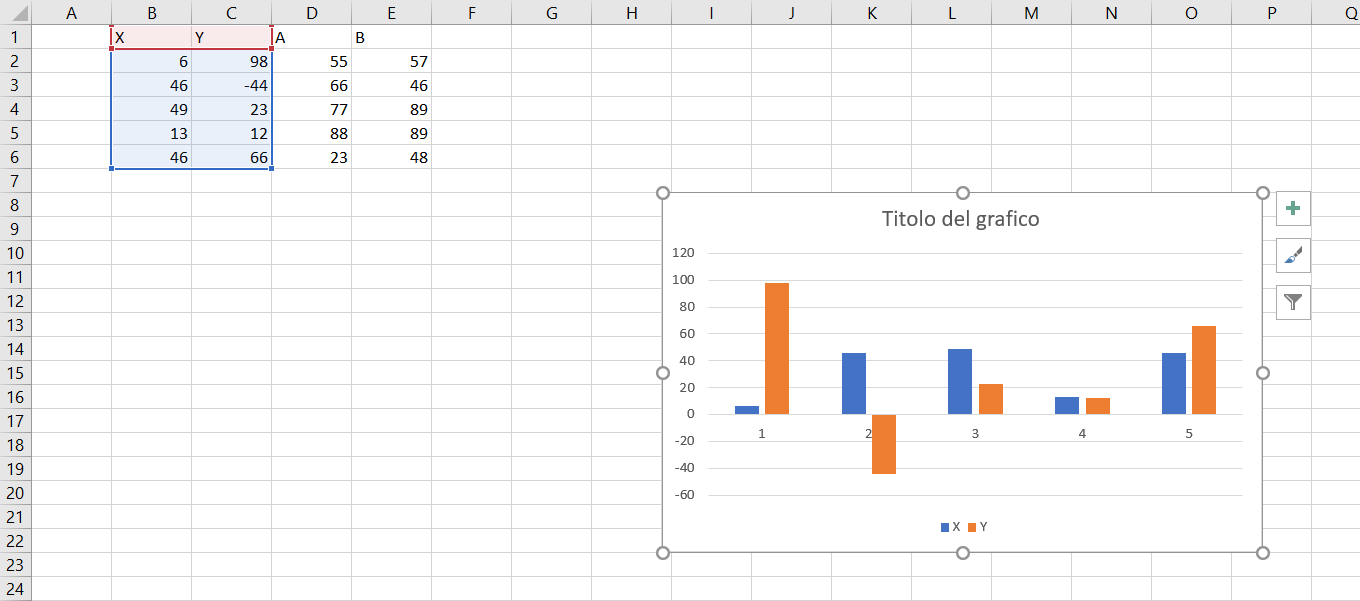
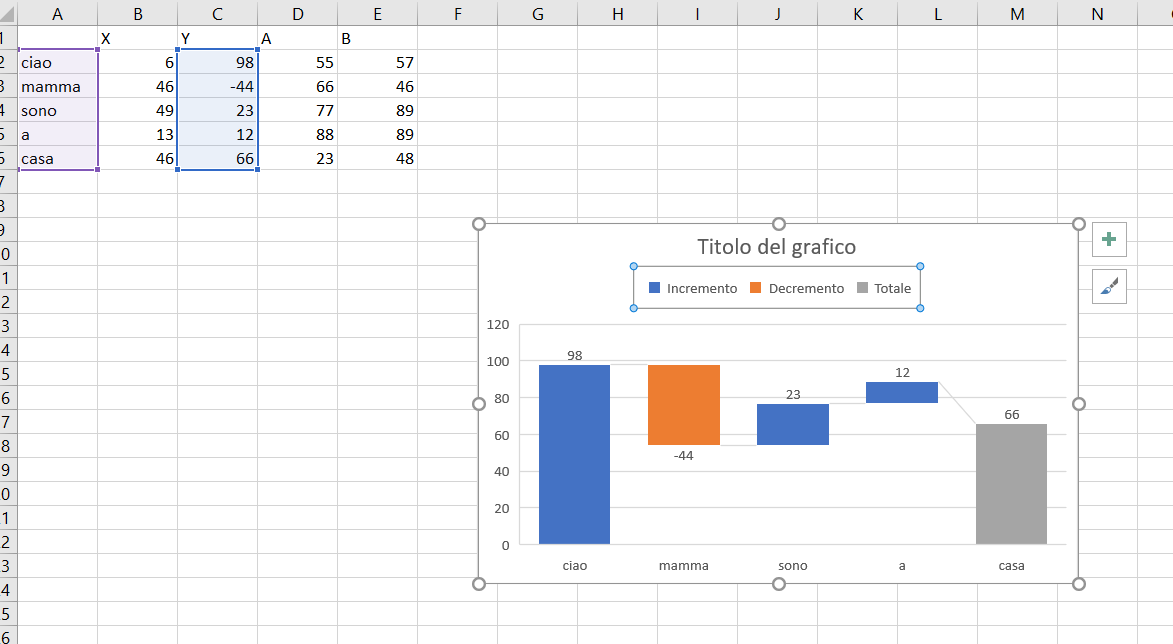
Devono essere inclusi almeno 5 tipi di grafici (Check-This) :

* Grafici Istogrami



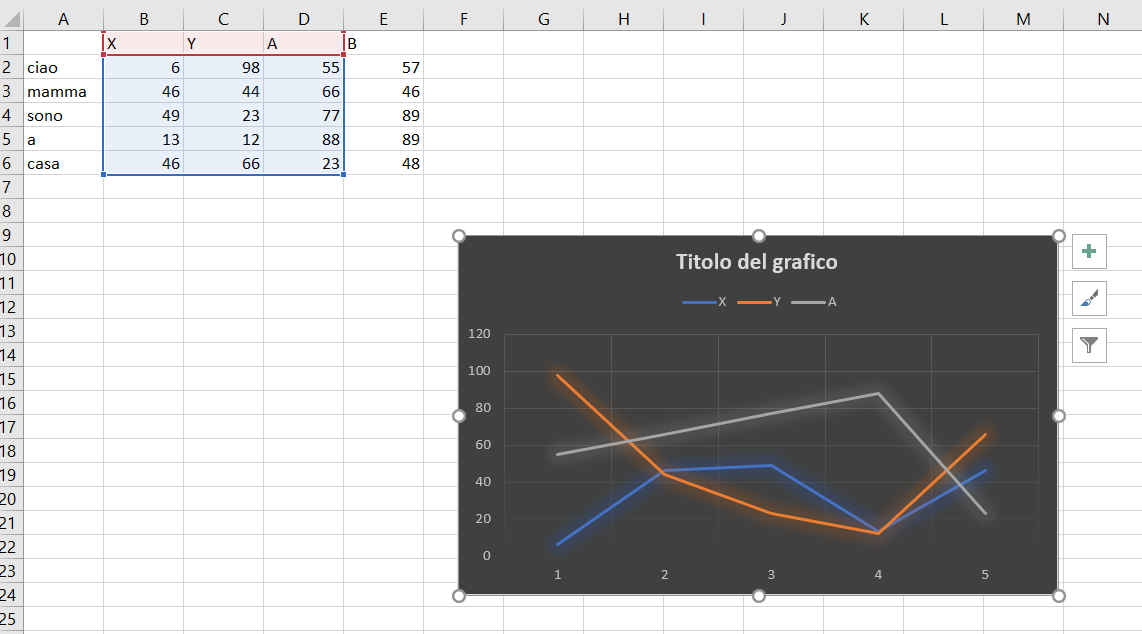
Accettano minimo un array di dati, nell’asse X ci vanno le identificazioni dell’array (nomi) e nell’asse Y ci vanno gli elementi dell’array.

* Grafici Azionari A Cascata



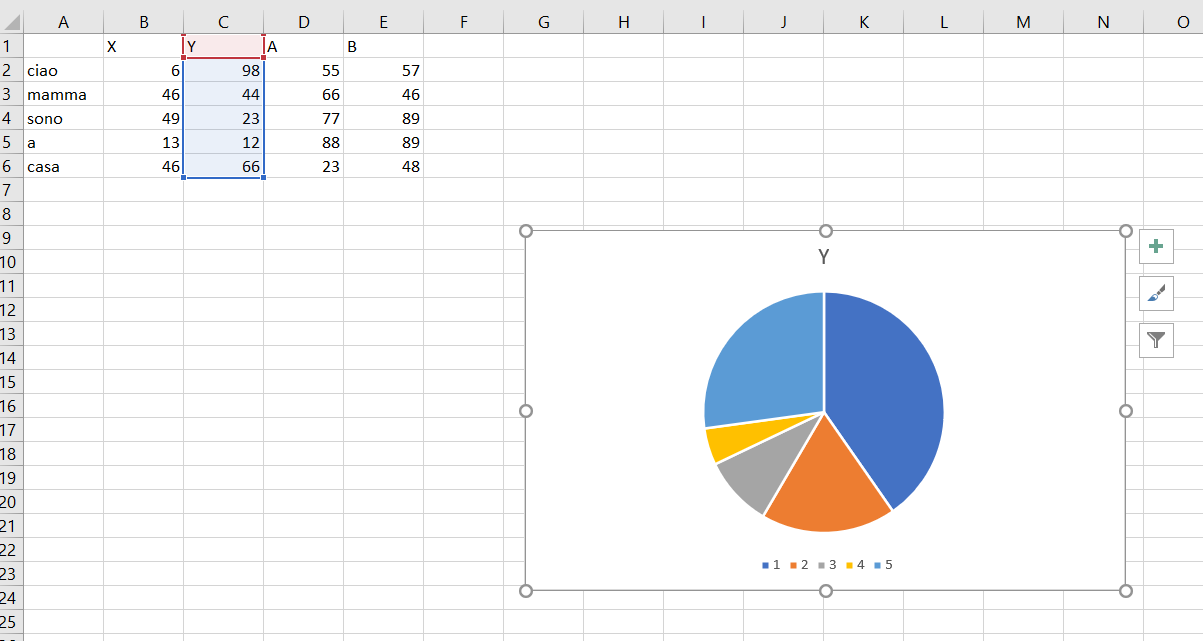
Accettano **solo** un array di dati (e un array di etichette), ci deve essere un campo separato per indicare quali sono i totali (o sub-totali).

* Grafici A Linee



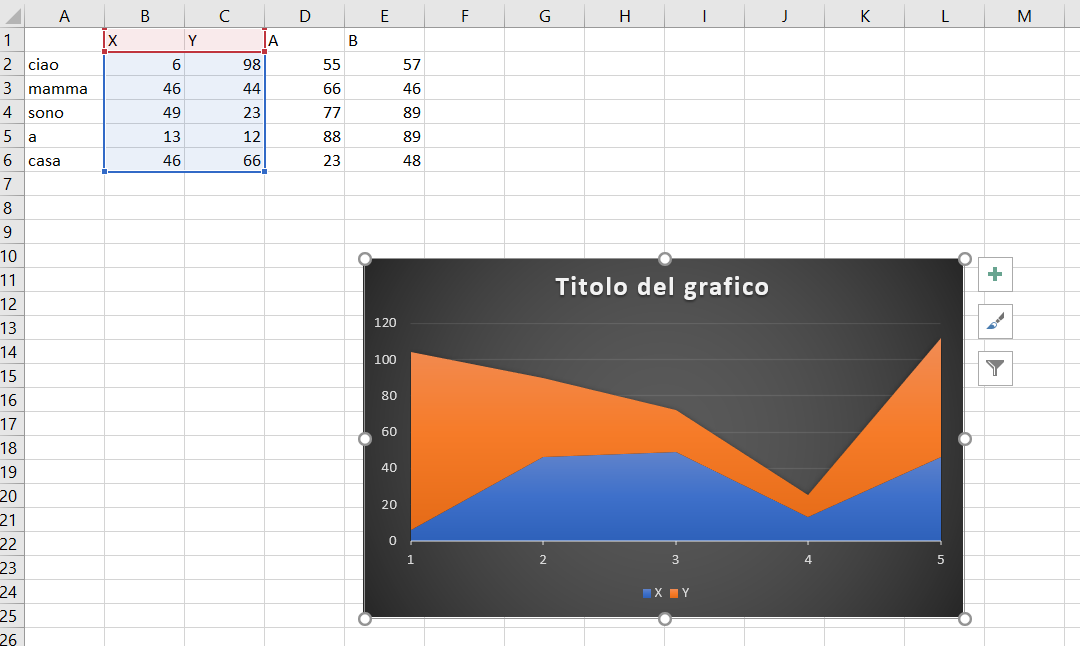
Accettano minimo un array di dati, nell’asse X ci vanno delle etichette e nell’asse Y ci vanno i dati dell’array.

* Grafico A Torta



Accettano **solo** un array di dati (e un array di etichette). Viene fatta la somma di tutti i dati (esempio, 98+44+23+12+66=243), poi viene fatta una proporzione per trovare la percentuale della fetta (243:100=98:x 🡪 (100\*98)/243 = 40% , fetta blu in immagine).

* Grafico Ad Area



Accettano minimo un array di dati. Nell’asse X ci vanno le etichette e nell’asse Y ci vanno i dati. Per il primo array dati (quello blu) viene disegnata l’area mettendo i suoi dati in Y, mentre per il secondo array (quello arancione) viene disegnata l’area mettendo in Y i suoi dati sommati a quelli dell’array precedente. (Esempio, l’array blu all’inizio parte da 0 a 6 mentre l’array arancione parte da 6 a (6+98)=104).

PASSAGGI PROGGETTO

* Visualizzazione GUI Basic dei Charts
* Pensare all’polimorfismo delle classi

1. Tenere conto della robustezza
2. Tenere conto dell’estensibilità (mediante polimorfismo)
3. Tenere conto della efficienza

* Pensare al salvataggio dei dati in file in formato JSON
* Pensare ad altre robe aggiuntive (tipo cambio colore grafo, grandezza dei font nei grafici ecc.)

Superclasse Grafici generici

* Classe Grafici di confronto
* Classe Grafici di andamento

Superclasse Dati generici (non sono sicuro)

* Dati da tastiera
* Dati da file

Superclasse modello

* Classe modello per il grafico (ciascun opzione)

Classe modello per la tabella (uno per ciascun opzione, non hanno superclasse in comune fatta da noi dato che ereditano una classe da Qt)

**Ragionamenti utili al polimorfismo**

La **main window** è una classe base che dovrebbe rappresentare solo la cornice del software.

Si suppone che ci possano essere in futuro più specialised window che permettono di estendere il nostro software a un’interfaccia diversa volendo sempre mantenere quello vecchio (senza doverlo modificare), si guardi l’esempio nel .png in qui si parla di “all-in” e “only one” (noi implementeremmo solo il caso only one).

**Select graphic** è una nuova classe da creare che corrisponde alla nostra specialised window.

**-------------------------------**

La **create chart widget** è una classe base che rappresenta tutti quegli elementi che accomunano un grafico (si pensi agli esempi delle classi shape, square, circle,…).

**Barchart widget, linechart widget, piechart widget, …**, sono classi che si estendono da create chart widget, modificando nel caso serva I vari metodi (bisogna ancora capire se tramite override o virtual).

**-------------------------------**

La **model** è una classe model base, tecnicamente dovrà essere vuota in modo tale da essere implementata dalle seguenti sottoclassi (ragionamento simile a quello tra la “main window” che possiamo chiamarla view base e le sue sottoclassi).

**ChartModel** è una nuova classe che estende model. In questa si cercherà di creare una rappresentazione universale di tutti i dati storati dai nostri 5 grafici. Se poi dovranno essere fatti dei cambiamenti questi verranno fatti in altre sue sottoclassi.

Un esempio che possa permettere di capire meglio quello scritto sopra…; Si supponga che la struttura dati usata in una pieChartModel sia una map (che mappa delle stringhe “eticchette” con dei valori int “i dati”). In chartModel la struttura dati sarà composta universalmente da una lista di stringhe e una lista di interi. Nella sottoclasse pieChartModel poi queste due liste verranno convertite in un'unica map.

**BarChartModel, LineChartModel, PieChartModel, …**, sono classi che si estendono da ChartModel, modificando nel caso serva le strutture dati (Ora sono stanco e non mi è ancora ben chiara l’idea su come verrà implementata la modifica, mi informerò domani).