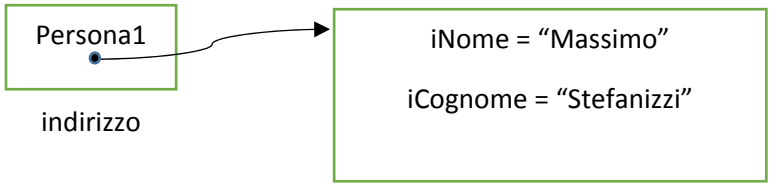
CONSIDERAZIONI SULLE CLASSI.

Abbiamo visto in precedenza cosa s'intende per classe e oggetto. Vediamo in dettaglio cosa succede quando istanziamo un oggetto.

In Visual C# (ma in tutti i linguaggi di programmazione OOP) quando si istanzia un oggetto, il nome dell'oggetto contiene, in realtà, l'indirizzo dell'area di memoria che contiene i dati ad esso associati.

Il nome dell'oggetto, per dirlo più correttamente, è un'aria di memoria che contiene l'indirizzo dell'aria di memoria che contiene i dati.

Ad esempio supponiamo di aver creato la classe Persona i cui attributi sono iNome, iCognome, con un metodo costruttore e/o le proprietà per istanziare un oggetto. Quando istanziamo l'oggetto Persona1 (in Visual C# tramite l'istruzione **Persona Persona1 = new Persona("Massimo", "Stefanizzi");**) accade:



Area di memoria contenente i dati dell'oggetto **Persona1**

Quindi, quando uso l'istruzione **NomeClasse NomeOggetto**; creo soltanto un'area di memoria destinata a contenere l'indirizzo dell'area di memoria che conterrà i dati. Quest'area di memoria si crea quando utilizzo il comando **new** più il costruttore.

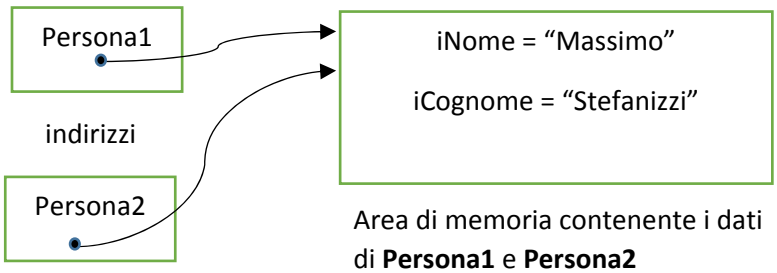
Consideriamo il codice seguente:

```
Persona Persona1 = new Persona("Massimo","Stefanizzi");

Persona Persona2;

Persona2 = Persona1;
```

ciò che accade lo illustriamo di seguito:

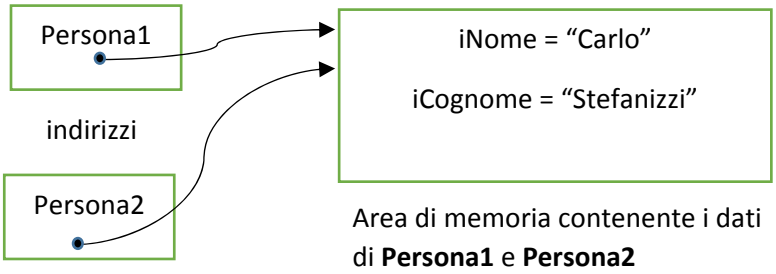


Area di memoria contenente i dati di **Persona1** e **Persona2**

Supponendo che nella classe persona siano presenti delle proprietà **SET** (supponiamo la proprietà ***Nome*** per modificare il nome di un oggetto), utilizzando l'istruzione:

Persona1.Nome = "Carlo";

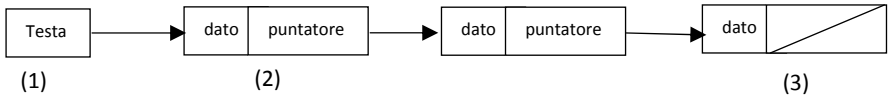
otteniamo:



Quindi se visualizzassimo il nome di Persona2, anche se abbiamo agito solo su Persona1, anche questo risulterebbe “Carlo”.

LISTE A PUNTATORI.

Una lista a puntatori è una struttura ‘dinamica’ in cui ogni suo elemento è formato da almeno due campi: un campo **dati**, che contiene le informazioni, ed un campo **puntatore** che contiene l’indirizzo del prossimo elemento della lista. Il primo elemento della lista è detto Testa, l’ultimo Coda caratterizzato dal campo puntatore contenente ‘null’ ossia non punta a nessun altro elemento. I campi puntatori consentono il collegamento tra i vari elementi della lista non necessariamente posti in aree contigue della memoria. Tale struttura è detta dinamica poiché non si conosce a priori il numero di elementi che costituiranno la lista. Ovviamente una tale struttura dati è una struttura sequenziale, nel senso che per esaminare la lista si deve scorrerla partendo dalla testa; non è quindi possibile accedere ad un qualsiasi elemento in modo ‘casuale’, come succedeva ad esempio, con i vettori.



(1) Puntatore al primo elemento della lista;
(2) Elemento di testa;
(3) Elemento di coda.

I passi per creare una lista sono:

(1) creazione del primo elemento;
(2) creazione del secondo elemento e inserimento in testa (o in coda) all’elemento creato in precedenza;
(3) ripetere il punto (2) sino al completamento della lista.

L’esempio guida, inserito nella pagina, mostra come creare una lista a puntatori.

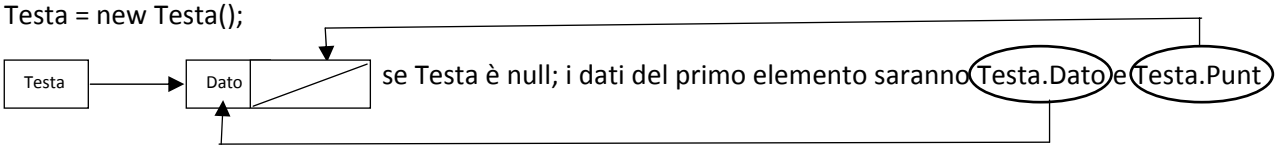
Si comincia con creare la classe Elemento (io l’ho chiamata Nodo) che ha gli attributi **Dato** di tipo stringa (sarà del tipo desiderato dall’utente) e l’attributo **Punt** di tipo Nodo entrambi pubblici. Notate l’attributo Punt: questo attributo è pronto per contenere un indirizzo. Infatti se avete letto con attenzione il capitolo precedente, quando si dichiara un oggetto (senza l’uguaglianza a new) in realtà non facciamo altro che predisporre un area destinata a contenere l’indirizzo dell’area di memoria con i dati: **public Nodo Punt** altro non è che la dichiarazione di un oggetto della classe Nodo chiamato Punt.

Stessa cosa quando dichiariamo l'oggetto di tipo Nodo **Testa** (di modulo) nella Form. Testa conterrà il l'indirizzo all'area di memoria con i dati dell'oggetto (Dato e Punt) ossia al primo elemento della lista. Gli elementi successivi, inseriti in testa alla lista, verranno agganciati secondo lo schema grafico seguente:

Nodo Testa;

è stato dichiarato l'oggetto Testa che conterrà l'indirizzo dell'area di memoria.

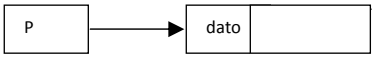
Testa = new Testa();



se Testa è null; i dati del primo elemento saranno Testa.Dato e Testa.Punt

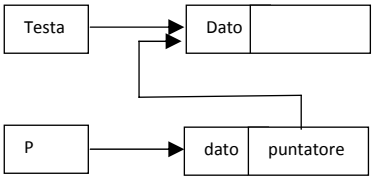
altrimenti

Nodo P = new Nodo();

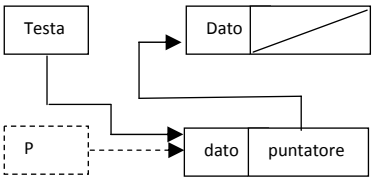


i dati di questo elemento saranno P.Dato e P.Punt, come sopra.

P.Punt = Testa;



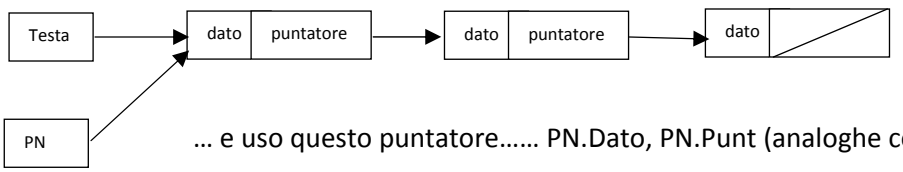
Testa = P;



Il vecchio puntatore P finirà nella Garbage Collection.

Il puntatore Testa non deve mai essere modificato pena la perdita di tutta o parte della lista. Infatti per visualizzare la lista nella listBox creo un altro puntatore PN nel quale copio l'indirizzo del primo elemento della lista contenuto in Testa.

Nodo PN = Testa;



… e uso questo puntatore…… PN.Dato, PN.Punt (analoghe considerazioni di sopra).