LISTE CON PRIORITA' IMPLEMENTATE TRAMITE STRUTTURE COLLEGATE

Liste con priorità

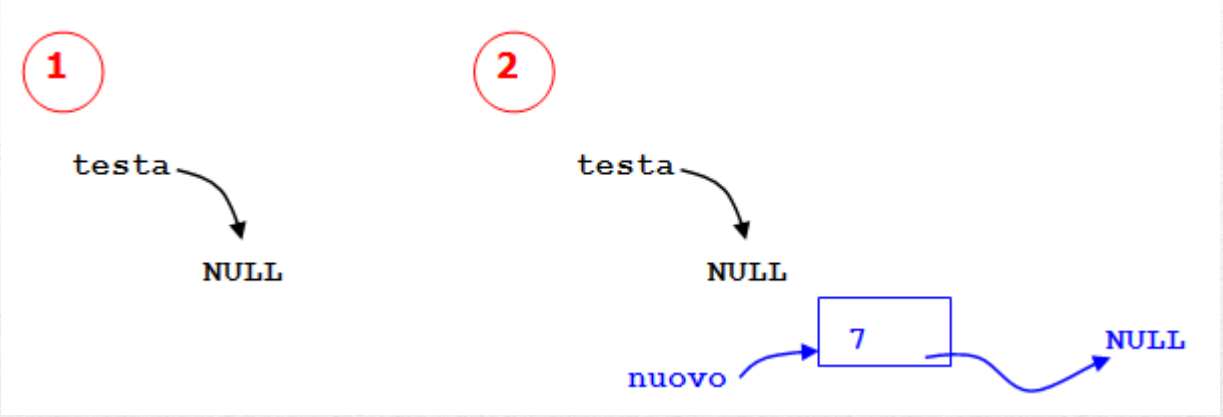
Le liste con **priorità** sono gestite come le liste semplici, solo che gli elementi all'interno della lista vengono mantenuti **ordinati** rispetto a un qualche valore che ne indica la priorità nei confronti degli altri elementi.

Immaginiamo per esempio l'accettazione di un pronto soccorso: i vari pazienti saranno visitati (elaborati) in ordine sia di arrivo che - soprattutto - di gravità, quindi un paziente grave sarà visitato prima di uno meno grave anche se arrivato in un momento successivo al pronto soccorso.

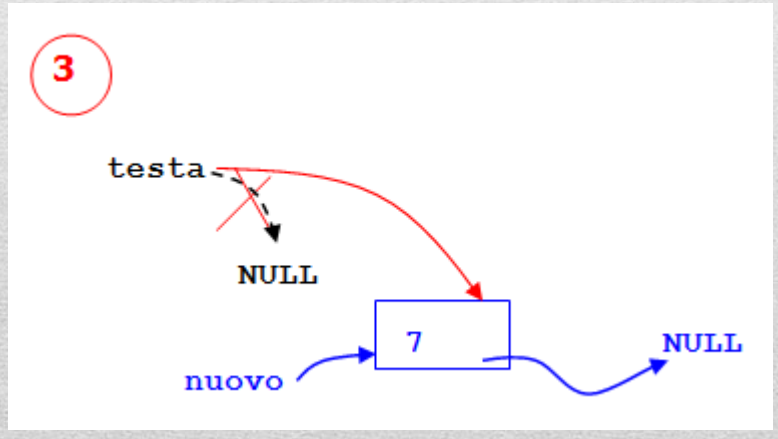
La struttura dati necessaria per gestire una lista con priorità è la stessa delle liste semplici:

```
typedef int TipoElemLista;
struct StructLista {
    TipoElemLista info;
    struct StructLista *next;
};

typedef struct StructLista TipoNodoLista;
typedef TipoNodoLista *TipoLista;

TipoLista testa = NULL;
```

Inserimento

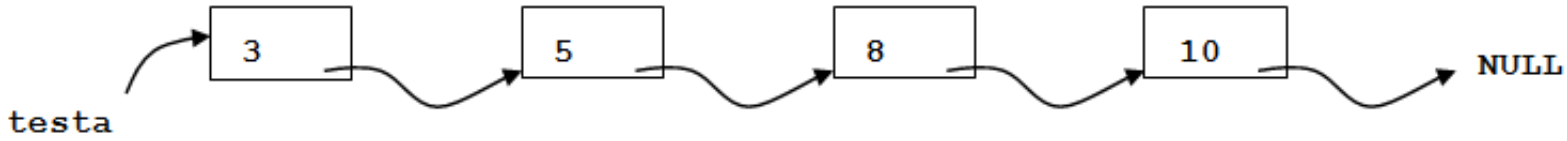



```
nuovo = (TipoLista))malloc(sizeof(TipoNodoLista));
//controlli
```

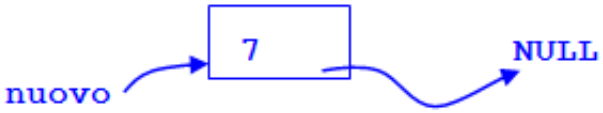



```
testa = nuovo;
```

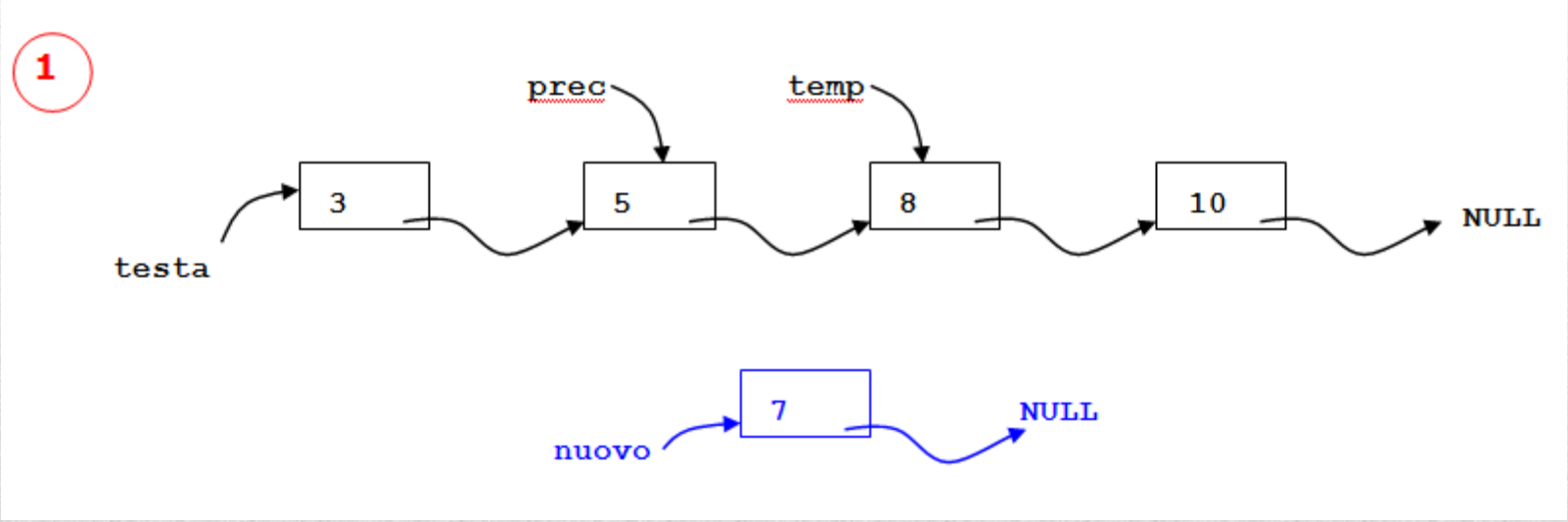
Inserimento



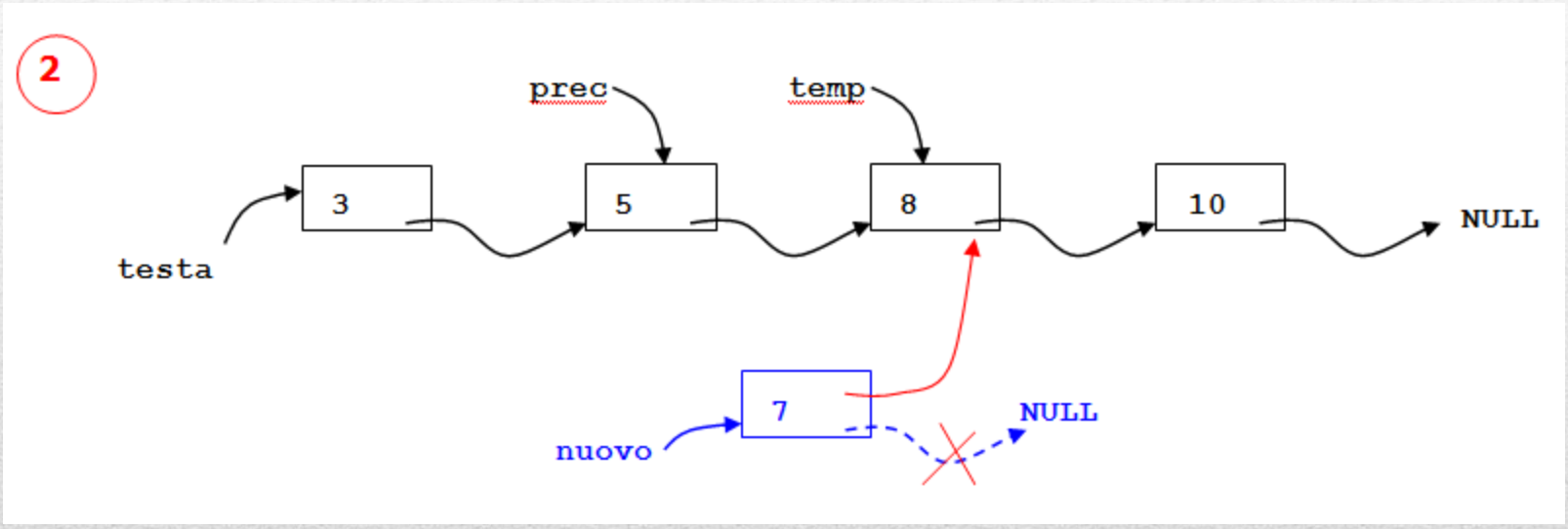
Se il nuovo elemento da inserire ha come ordine il numero 7



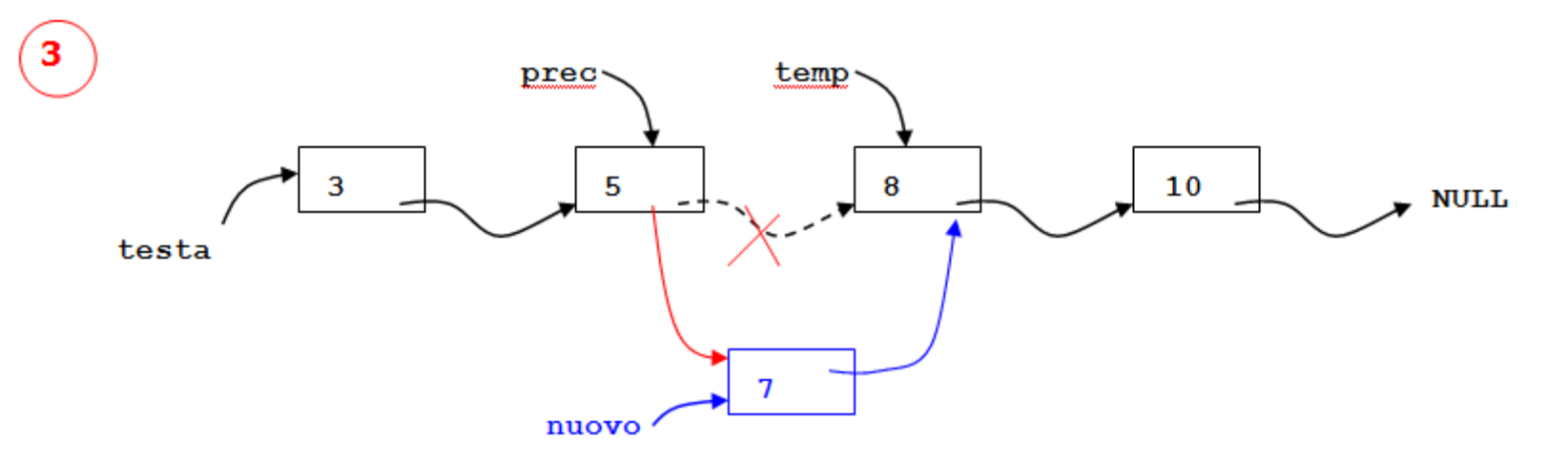
Inserimento



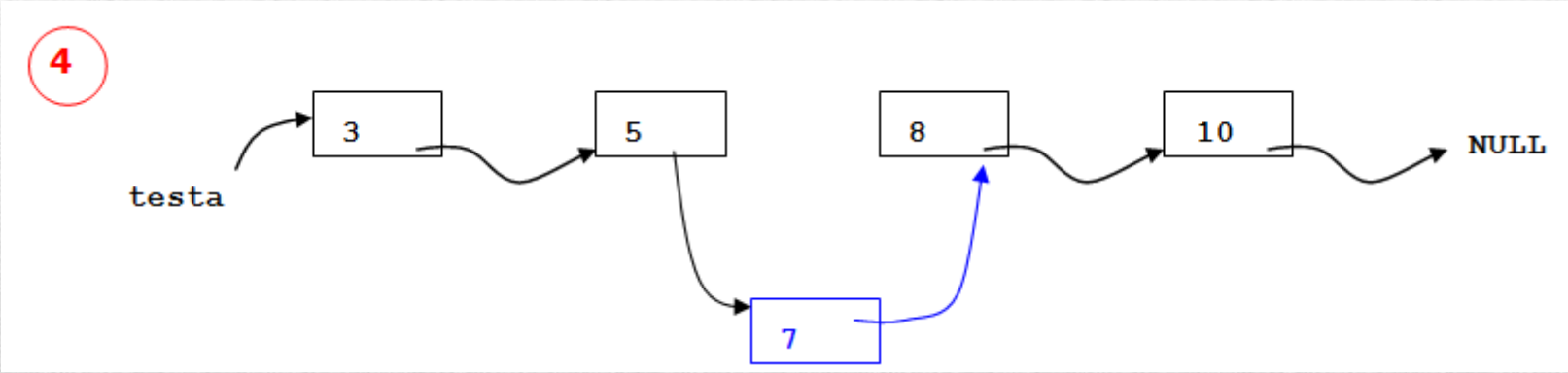
Inserimento



Inserimento



Inserimento



Inserimento

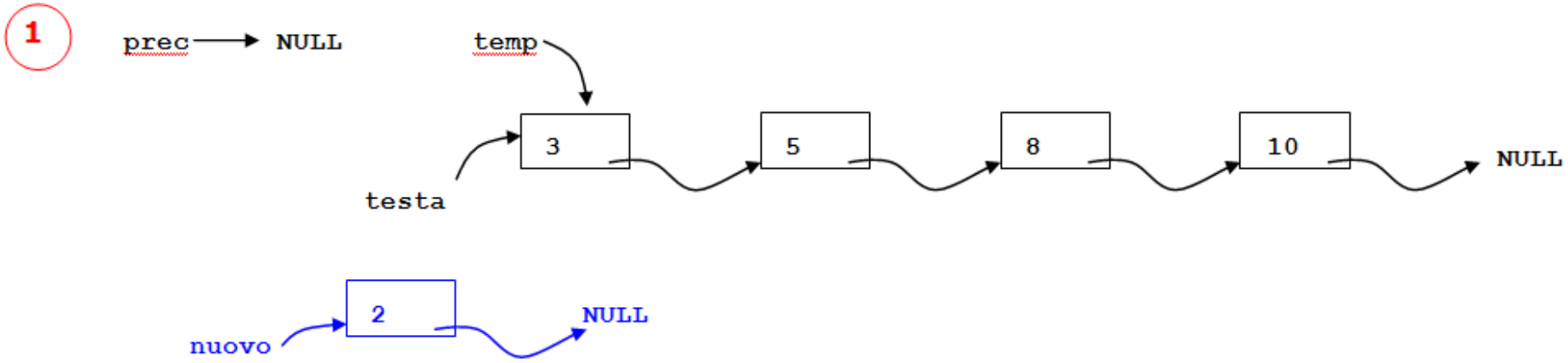
Per inserire la nuova cella dovremo scorrere la lista fino ad arrivare alla prima cella con valore maggiore rispetto a quella che deve essere inserita. Dato che la lista è ordinata in maniera crescente, il primo elemento maggiore sarà anche quello che dovrà stare immediatamente dopo al nuovo elemento, una volta effettuato l'inserimento.

Inserimento

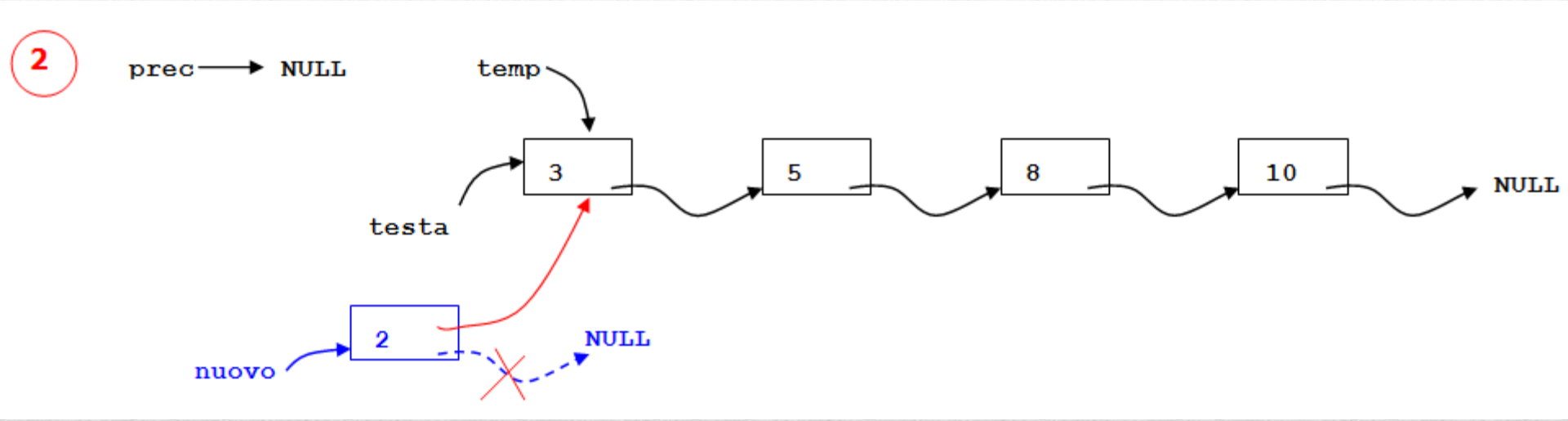
```
nuovo= (TipoLista)malloc(sizeof(TipoNodoLista));
//controlli
nuovo->info=ValoreDaInserire;
trovato = FALSE;
if (testa == NULL)
  testa=nuovo;
else
{
  prec=NULL;
  temp=testa;
  while (temp != NULL && !trovato)
  {
    if (temp->info > nuovo->info)       //trovata cella con valore più grande
    {
      nuovo->next=temp;
      if (prec == NULL)                 //l'elemento è da inserire in testa
        testa=nuovo;
      else
        prec->next=nuovo;
      trovato=TRUE;
    }
    prec=temp;
    temp=temp->next;
  }
}
```

Inserimento

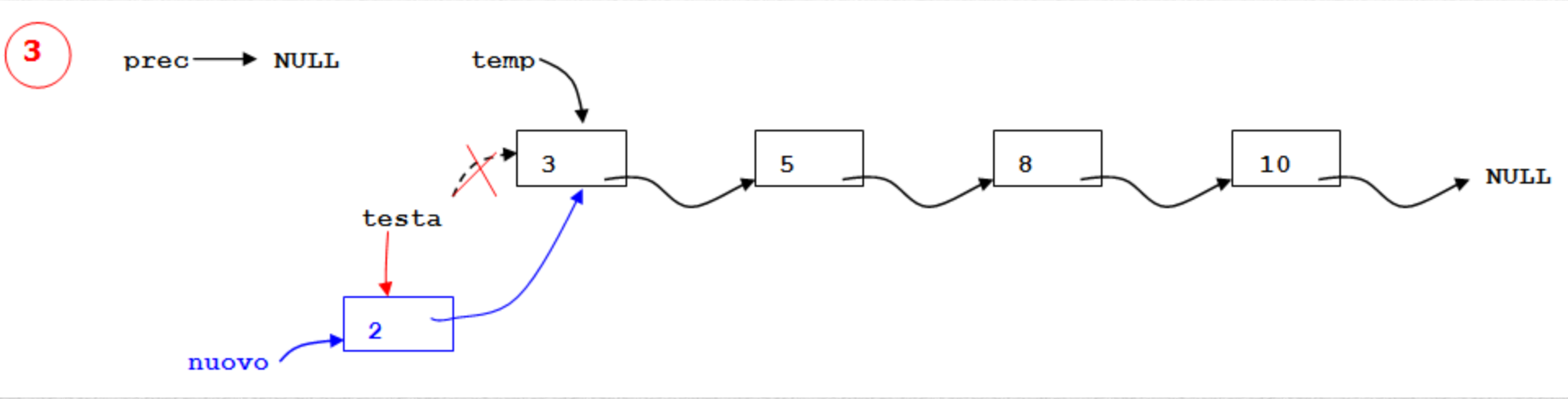
Se l'elemento deve essere inserito in testa:



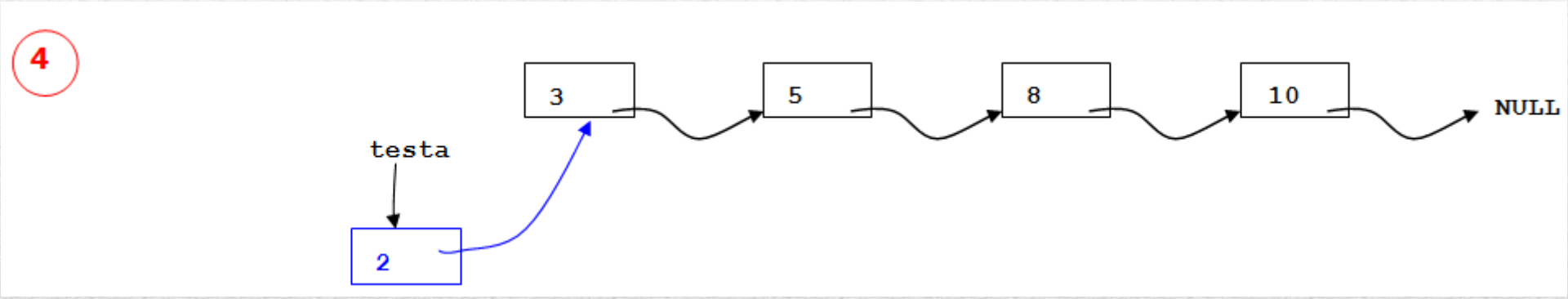
Inserimento



Inserimento

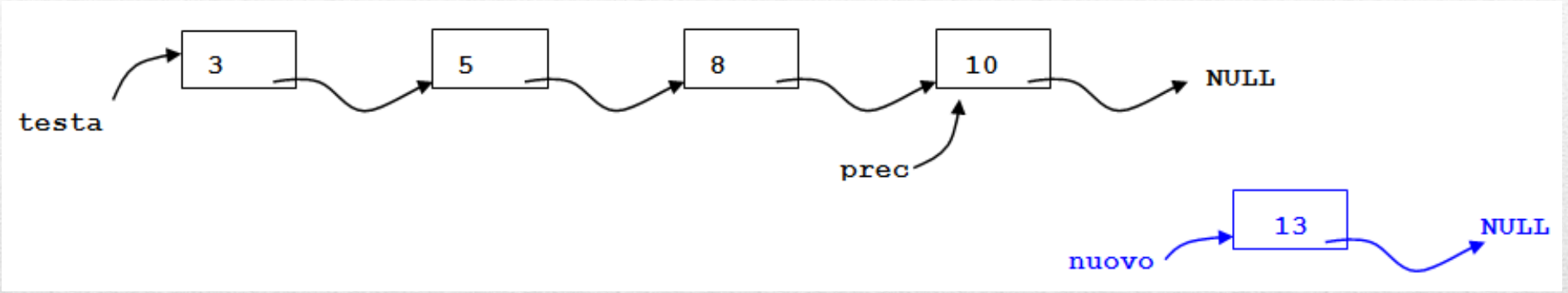


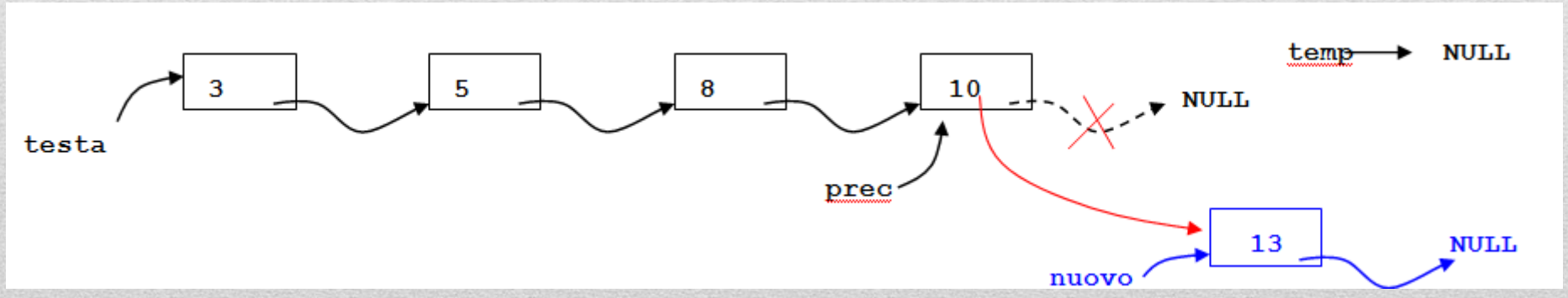
Inserimento



Inserimento

Se l'elemento deve essere inserito in coda, l'algoritmo descritto sopra non funziona in quanto all'interno del ciclo while non si troverà mai una cella più grande rispetto a quella da inserire:



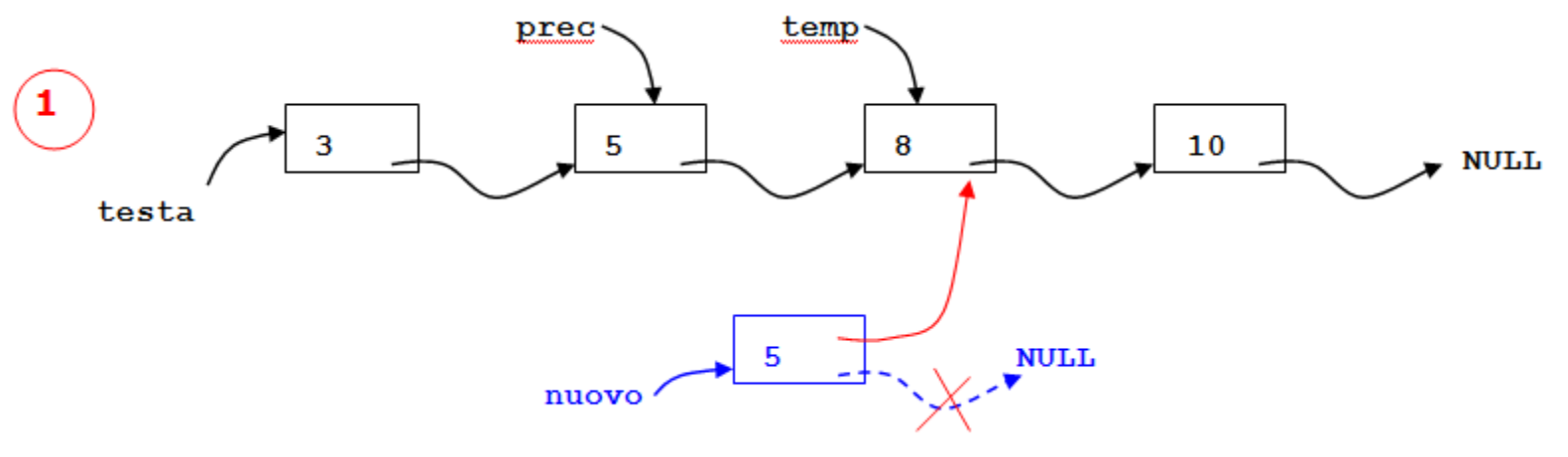


Inserimento

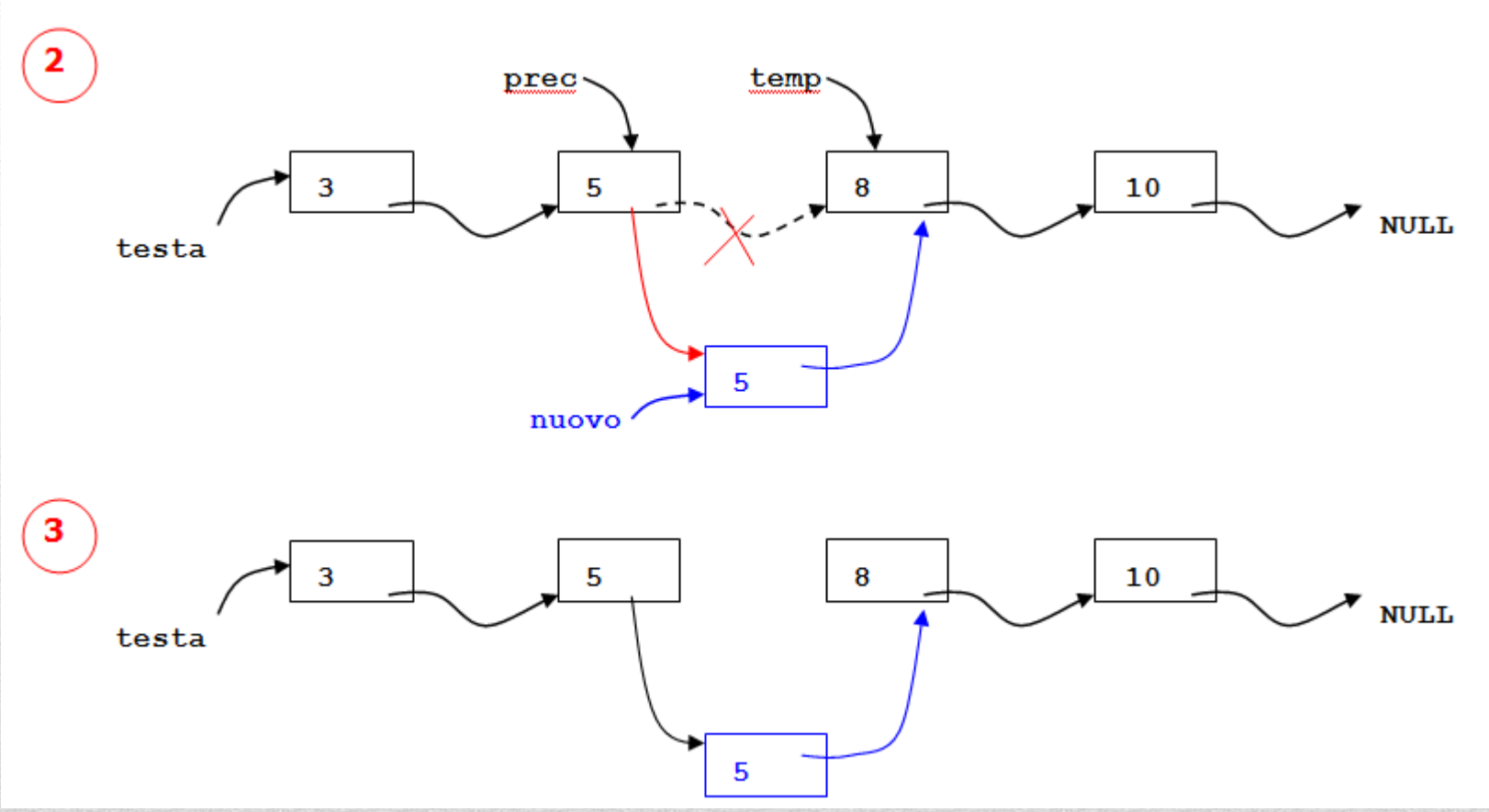
```
nuovo= (TipoLista)malloc(sizeof(TipoNodoLista));
//controlli
nuovo->info=ValoreDaInserire;
trovato = FALSE;
if (testa == NULL)
  testa=nuovo;
else
{
  prec=NULL;
  temp=testa;
  while (temp != NULL && !trovato)
  {
    if (temp->info > nuovo->info)      //trovata cella con valore più grande
    {
      nuovo->next=temp;
      if (prec == NULL)                //l'elemento è da inserire in testa
        testa=nuovo;
      else
        prec->next=nuovo;
      trovato=TRUE;
    }
    prec=temp;
    temp=temp->next;
  }
  if (temp == NULL)                    //il nuovo elemento non è stato inserito e siamo in fondo alla lista
    prec->next=nuovo;
}
```

Inserimento

Nel caso in cui il valore della nuova cella da inserire sia già presente all'interno della lista, il programma presentato inserisce la nuova cella dopo quella con valore uguale:



Inserimento



Come fare se la nuova cella dovrà essere inserita prima di quella con il valore uguale?