Fpt University

FPT POLYTECHNIC

Bài 7:

DANH SÁCH LIÊN KẾT (LINKED LIST) VÀ TẬP HỢP (SET)

Nhắc lại bài cũ

- Tìm hiểu 3 cấu trúc dữ liệu đặc biệt: Ngăn xếp (Stack), Hàng đợi (Queue) và Cây (Tree):
  - Khái niệm
  - Cách cài đặt trong VB.Net
  - Các thao tác cơ bản trên các cấu trúc dữ liệu

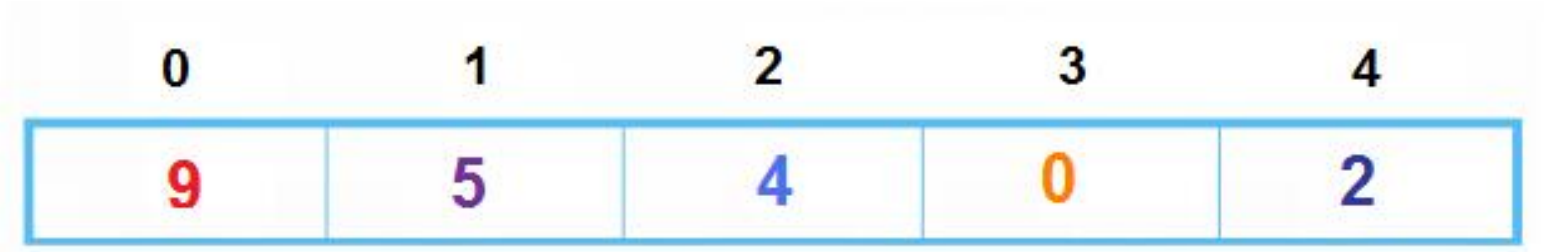
FPT POLYTECHNIC

Mục tiêu bài học hôm nay

- Tìm hiểu về danh sách liên kết (Linked List):
  - Khái niệm danh sách liên kết
  - Các thao tác trên danh sách liên kết
- Tìm hiểu về Set (tập hợp):
  - Định nghĩa Set
  - Các đặc trưng
  - Cách cài đặt Set bằng VB.Net

Khái niệm Danh sách liên kết

- Mảng là một hình thức liên kết ngầm:
  - Các phần tử trong mảng truyền thống được cấp phát vùng nhớ một cách liên tiếp nhau

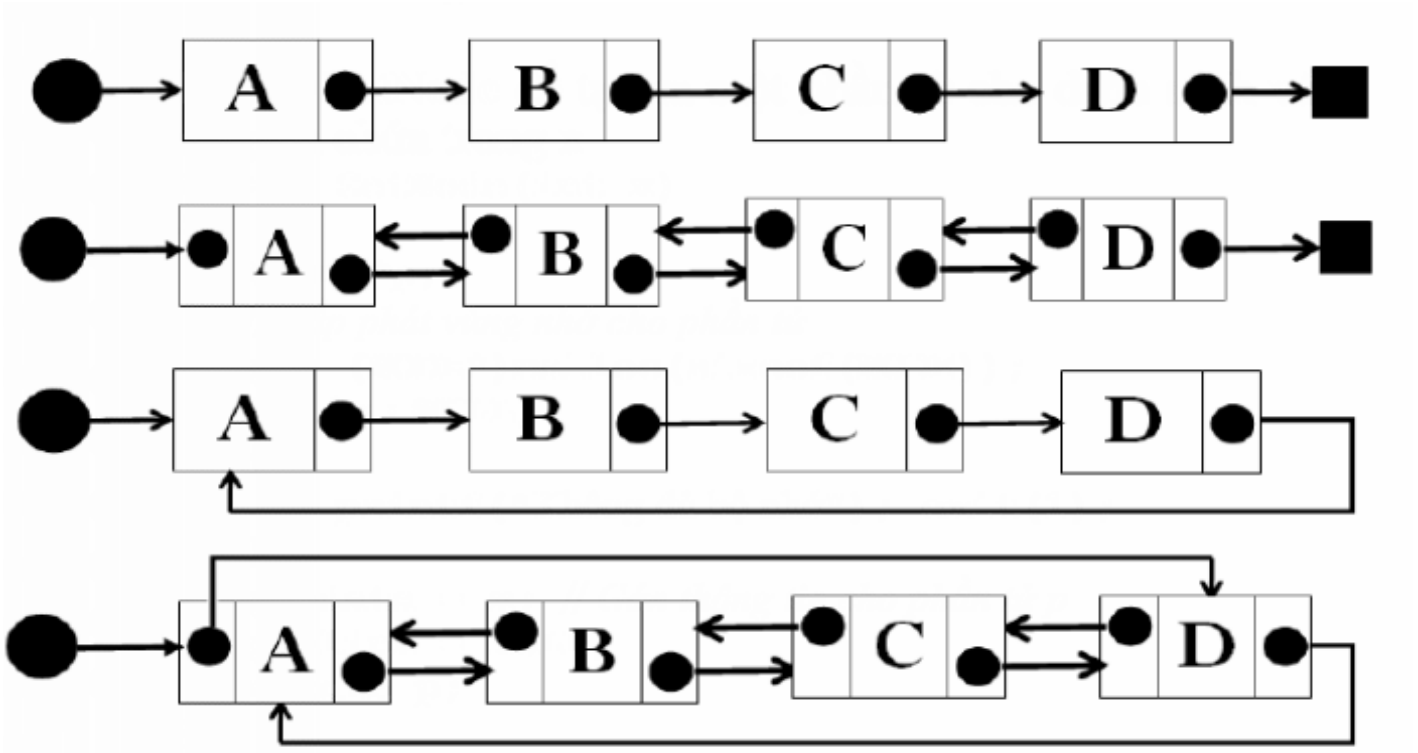


FPT POLYTECHNIC

Khái niệm danh sách liên kết

- Danh sách liên kết cũng gồm nhiều phần tử như mảng, nhưng có sự khác biệt cơ bản là các phần tử không nằm liên tiếp nhau trong bộ nhớ, mà dùng liên kết để móc nối với nhau.
- Có nhiều loại danh sách liên kết:
  - Danh sách liên kết đơn
  - Danh sách liên kết kép
  - Danh sách liên kết vòng
  - ...
- Trong bài này ta tìm hiểu kĩ về danh sách liên kết đơn

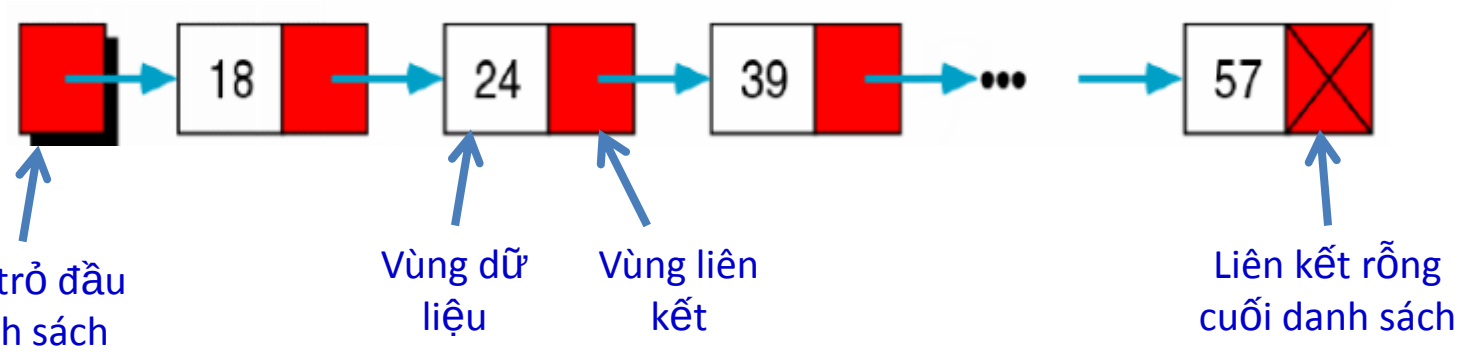
Khái niệm Danh sách liên kết



Hình mô tả 4 kiểu danh sách liên kết

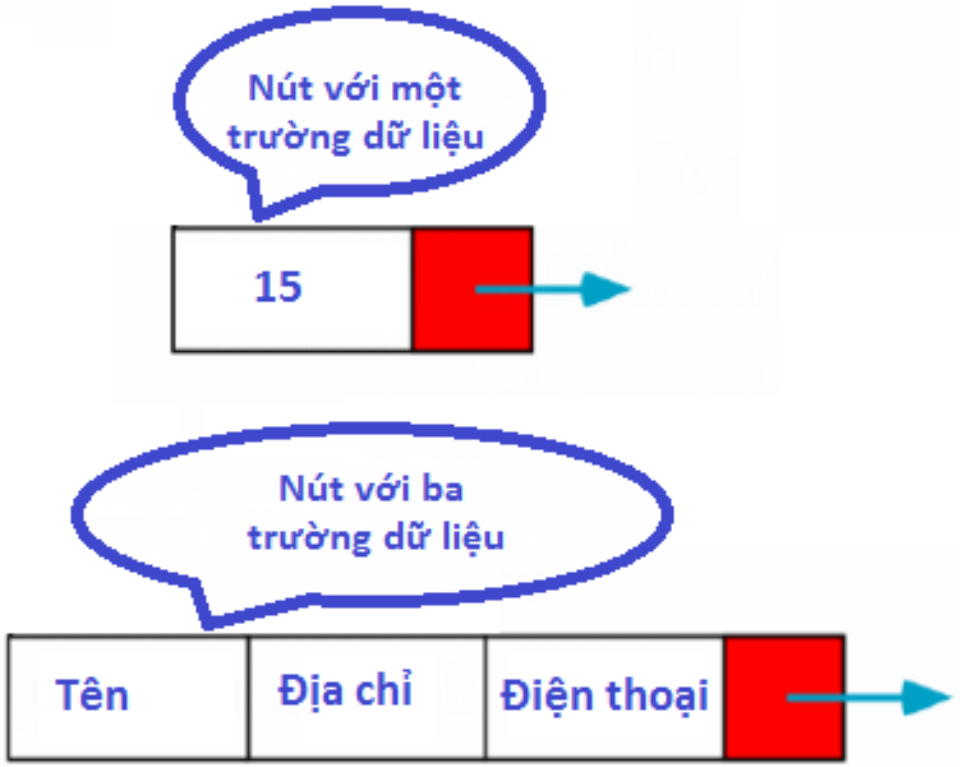
Khái niệm danh sách liên kết

- Danh sách liên kết đơn là danh sách gồm nhiều nút móc nối với nhau.
- Mỗi nút gồm:
  - Trường dữ liệu lưu trữ các giá trị của nút.
  - Trường lưu trữ liên kết
- Danh sách liên kết đơn cần có một con trỏ đầu danh sách (head) trỏ đến nút đầu tiên, các nút còn lại trỏ tuần tự đến nút kế tiếp và một liên kết rỗng (null) ở nút cuối cùng (không trỏ đến đâu).



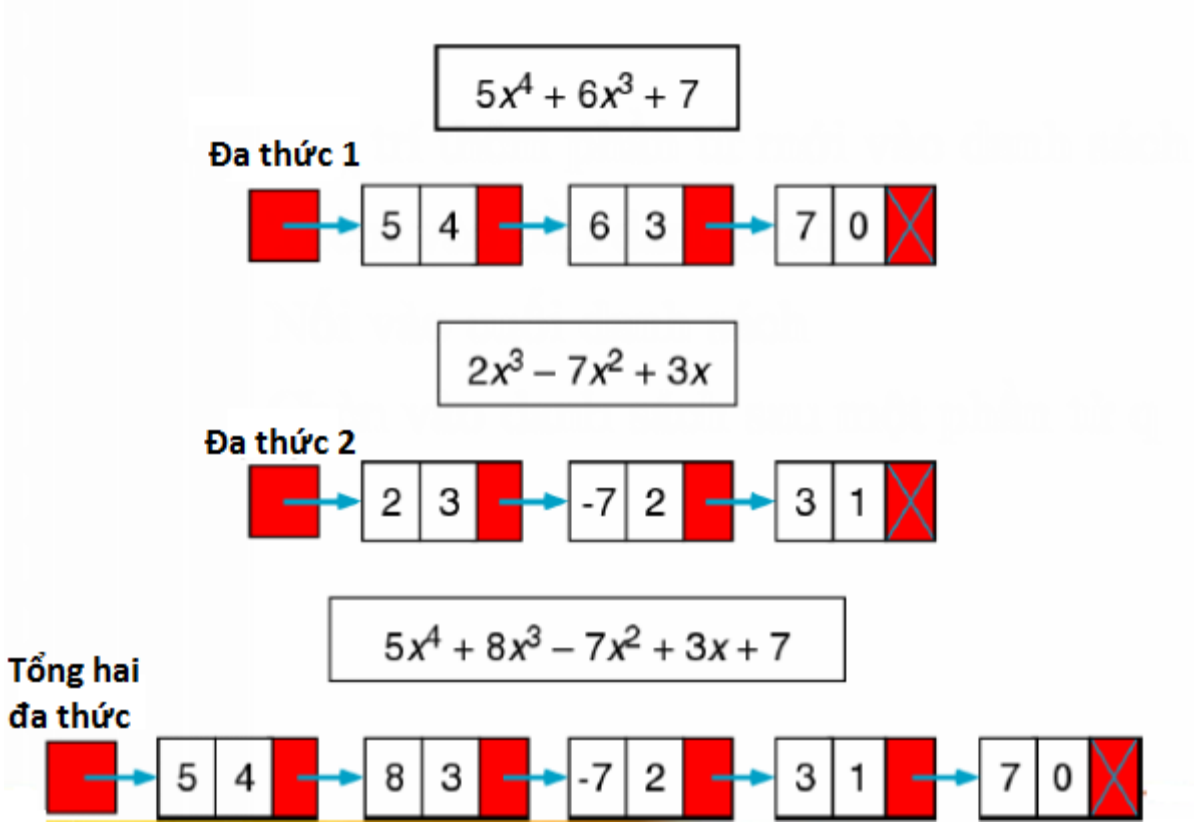
Khái niệm Danh sách liên kết

- Ví dụ:



Khái niệm Danh sách liên kết

- Ví dụ: dùng danh sách biểu diễn đa thức và tính tổng hai đa thức



Các thao tác trên Danh sách liên kết

- Tạo danh sách
  - Khai báo và khởi tạo danh sách liên kết
- Thêm phần tử:
  - Thêm vào đầu/cuối danh sách hoặc vào sau một phần tử cho trước
- Tìm kiếm phần tử trong danh sách:
  - Tìm kiếm phần tử có giá trị cho trước
  - Tìm kiếm phần tử đứng trước phần tử cho trước
- Xóa bỏ phần tử trong danh sách:
  - Xóa phần tử đầu/cuối danh sách
- Hiển thị nội dung của toàn bộ danh sách

Tạo danh sách liên kết

- Dùng lớp Node để tạo danh sách liên kết trong VB.Net: mỗi nút của danh sách gồm 2 trường
  - Trường Element lưu trữ thông tin của nút
  - Trường Link lưu trữ liên kết đến nút kế tiếp
- Khai báo và hàm tạo:

```
Public Class Node
    Public Element As Object
    Public Link As Node
    Public Sub New()
        Element = Nothing
        Link = Nothing
    End Sub
    Public Sub New(ByVal theElement As Object)
        Element = theElement
        Link = Nothing
    End Sub
```

Tìm kiếm phần tử trong Danh sách

- Thao tác tìm kiếm một phần tử trong danh sách:

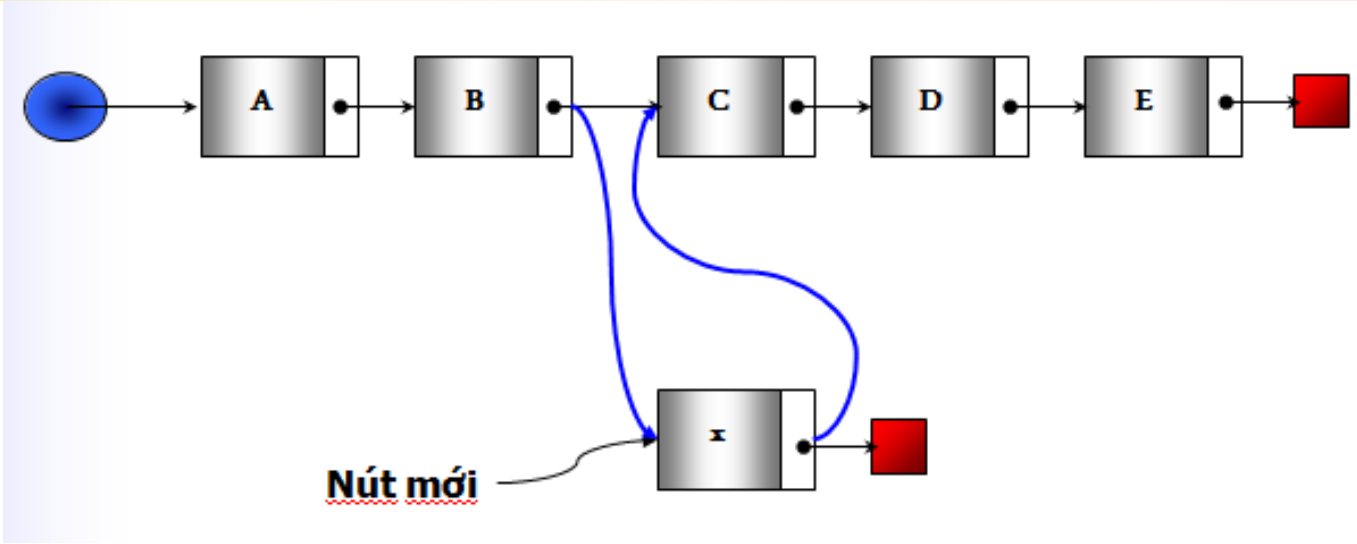
```
Private Function Find(ByVal item As Object) As Node
    Dim current As New Node()
    current = header
    While (current.Element <> item)
        current = current.Link
    End While
    Return current
End Function
```

Tìm kiếm phần tử trong Danh sách

- Tìm kiếm một phần tử trước một phần tử khác trong danh sách:

```
Private Function FindPrevious(ByVal x As Object) As Node
    Dim current As Node = header
    While (Not (current.Link Is Nothing) And _
    current.Link.element <> x)
        current = current.Link
    End While
    Return current
End Function
```

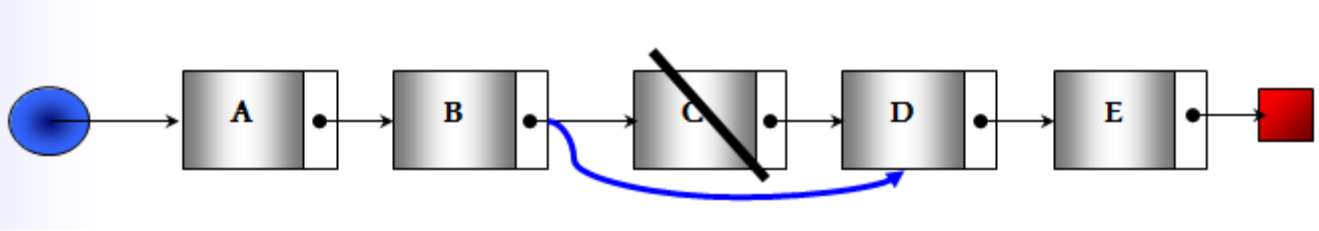
Thêm phần tử vào Danh sách




```
Public Sub Insert(ByVal newItem As Object, ByVal after
As Object)
    Dim current As New Node()
    Dim newnode As New Node(newItem)
    current = Find(after)
    newnode.Link = current.Link
    current.Link = newnode
End Sub
```

Xóa phần tử trong Danh sách

- Thao tác xóa một phần tử:




```
Public Sub Remove(ByVal x As Object)
    Dim p As Node = FindPrevious(x)
    If (Not (p.Link Is Nothing)) Then
        p.Link = p.Link.Link
    End If
End Sub
```

Truy xuất dữ liệu các phần tử trong Danh sách

- Hiển thị nội dung các phần tử trong danh sách

```
Public Sub PrintList()
    Dim current As New Node()
    current = header
    While (Not (current.Link Is Nothing))
        Console.WriteLine(current.Link.Element)
        current = current.Link
    End While
End Sub
```

Ưu điểm của Danh sách liên kết

- Các thao tác thêm, bớt các phần tử trong danh sách khá dễ dàng, chỉ cần thay đổi mối liên kết giữa các phần tử với nhau
- Kích thước danh sách được cấp phát tự động -> tiết kiệm bộ nhớ
- Số phần tử trong danh sách liên kết có thể tăng vô hạn, tùy thuộc vào kích thước bộ nhớ

FPT POLYTECHNIC

Hạn chế của Danh sách liên kết

- Việc truy xuất và tìm kiếm các phần tử mất nhiều thời gian hơn so với mảng bởi luôn luôn phải duyệt tuần tự qua các phần tử trong danh sách.
- Tốn bộ nhớ hơn mảng vì lưu trữ thông tin mỗi nút và thêm thông tin vùng liên kết.

Định nghĩa SET

- Định nghĩa: Set là tập hữu hạn các phần tử (thành viên) và có 2 tính chất:
  - Các phần tử không được sắp xếp theo thứ tự
  - Mỗi phần tử không được xuất hiện nhiều hơn 1 lần

- Khai báo một Set: *{phần tử 1, phần tử 2,…, phần tử n}*

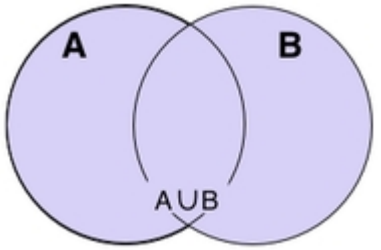
Ví dụ: {0,1,2,3,4,5,6,7,8,9}

Các định nghĩa khác

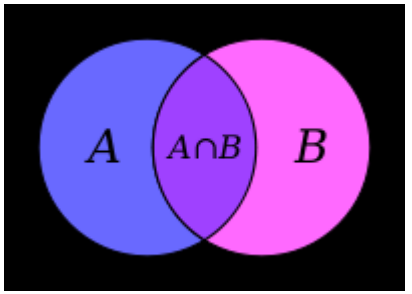
- Tập rỗng (empty set): không chứa phần tử nào
- Tập vũ trụ (universe set): chứa mọi phần tử có thể
- Tập bằng nhau: hai tập bằng nhau nếu chúng chứa các phần tử như nhau.
- Tập con: một tập gọi tập con của một tập khác nếu mọi phần tử của nó nằm trong tập kia.

Các phép toán trên Tập

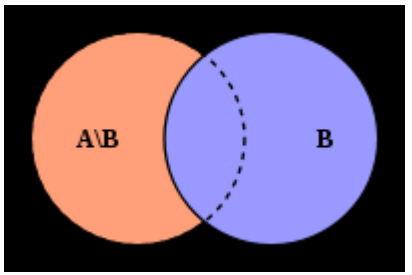
- Phép hợp (Union)
  - Hợp của A và B là tập gồm tất cả các phần tử thuộc ít nhất một trong hai tập A và B



- Phép giao (Intersection)
  - Giao của hai tập A và B là tập tất cả các phần tử vừa thuộc A, vừa thuộc B



- Phép Hiệu (Difference)
  - Hiệu của tập A với tập B là tập tất cả các phần tử thuộc A nhưng không thuộc B



Cài đặt Tập

- VB.Net cung cấp một số lớp cài đặt tập:
  - Hashtable
  - BitArray

Sử dụng lớp Hastable

- Ví dụ phương thức khởi tạo

```
Public Class CSet
        ‘ khai báo tập hợp ‘data’
        Private data As Hashtable
        Public Sub New()
                data = New Hashtable
        End Sub
End Class
```

Sử dụng lớp Hastable

- Các phương thức khác:
  - Add: Thêm phần tử trong tập
  - Remove: Xóa phần tử trong tập
  - Size: Đếm số lượng phần tử trong tập
  - Union: Thực hiện hợp hai tập
  - Intersection: Thực hiện phép giao hai tập
  - Subset: Tạo tập con của một tập khác
  - Difference : Thực hiện phép trừ hai tập

Sử dụng lớp Hastable

- Ví dụ thực hiện phương thức Add:

```
Public Sub Add(ByVal item As Object)
        ‘ Kiểm tra xem tập hợp đã chứa item chưa
        If Not (data.ContainsValue(item)) Then
                data.Add(Hash(item), item)
        End If
End Sub
```

Sử dụng lớp Hastable

- Ví dụ thực hiện phương thức Union:

```
Public Function Union(ByVal aSet As CSet) As CSet
  Dim tempSet As New CSet
  Dim hashObject As Object
  'Xây dựng 1 tập hợp
  For Each hashObject In data.Keys
      tempSet.Add(Me.data.item(hashObject))
  Next
  'Add các phần tử từ tập hợp 2 vào
  For Each hashObject In aSet.data.Keys
      If (Not (Me.data.ContainsKey(hashObject))) Then
       tempSet.Add(aSet.data.Item(hashObject))
      End If
  Next
  Return tempSet
End Function
```

Sử dụng lớp BitArray

- BitArray được sử dụng để lưu trữ tập số nguyên.
- Nguyên tắc lưu trữ: phần tử có giá trị N trong tập được biểu diễn bởi giá trị True tại vị trí N của tập.
  - Ví dụ: tập chứa 1 và 4 là: {True, False, False, True}

Sử dụng lớp BitArray

- Lợi ích của sử dụng lớp BitArray:
  - Không gian bộ nhớ để lưu trữ nhỏ
  - Các thao tác Union, Intersection, Difference được thực hiện bằng cách sử dụng các toán tử AND, OR, NOT với thời gian cài đặt nhanh hơn.

Tổng kết

- Danh sách liên kết gồm tập hợp các nút liên kết với nhau thông qua vùng liên kết.
- Dùng lớp Node để cài đặt danh sách liên kết trong VB.Net
- Một số thao tác cơ bản trên danh sách liên kết: thêm phần tử, xóa phần tử, tìm kiếm, duyệt tất cả các phần tử.
- Tập (set) gồm hữu hạn các phần tử chưa được sắp thứ tự và mỗi phần tử xuất hiện đúng một lần.
- Dùng lớp Hashtable hoặc lớp BitArray để cài đặt tập trong VB.Net.