2025 年 11 月 19 日（オンライン）

アルゴリズムとデータ構造 B 課題 5

線形リストの動作に関する理解を深めるため，基本的な処理の途中および終了後の情報を表示するように `InsertFront, Remove, InsertbyIndex, Display` 関数を変更する.

問題 1

`Remove` 関数において削除するノードが見つかるまでの間，また，`InsertbyIndex` 関数においてノードの挿入位置が見つかるまでの間，ポインタ `p, q` がどのノードを指しているか逐次表示されるように，両関数を変更せよ. 出力例として以下のようなものが考えられる.

出力例 1）`q` と `p` が同じノードを指す場合

```
削除するノードのデータを入力してください：100
i = 0
     0: 0x6000036c0090    100 0x6000036c0080 <- q, p
     1: 0x6000036c0080     90 0x6000036c0070
     2: 0x6000036c0070     80 0x6000036c0060
     3: 0x6000036c0060     70 0x6000036c0050
     4: 0x6000036c0050     60 0x6000036c0040
     5: 0x6000036c0040     50 0x6000036c0030
     6: 0x6000036c0030     40 0x6000036c0020
     7: 0x6000036c0020     30 0x6000036c0010
     8: 0x6000036c0010     20 0x6000036c0000
     9: 0x6000036c0000     10 0x0
```

出力例 2）`q` と `p` が別のノードを指す場合

```
i = 1
     0: 0x6000019e4090    100 0x6000019e4080 <- q
     1: 0x6000019e4080     90 0x6000019e4070 <- p
     2: 0x6000019e4070     80 0x6000019e4060
     3: 0x6000019e4060     70 0x6000019e4050
     4: 0x6000019e4050     60 0x6000019e4040
     5: 0x6000019e4040     50 0x6000019e4030
     6: 0x6000019e4030     40 0x6000019e4020
     7: 0x6000019e4020     30 0x6000019e4010
     8: 0x6000019e4010     20 0x6000019e4000
     9: 0x6000019e4000     10 0x0
```

問題 2

InsertFront, InsertbyIndex 関数において挿入されたノード，Remove 関数において削除されたノードの一つ前のノードが，それぞれどのノードなのか明確になるように，Display 関数を変更せよ．InsertFront ではポインタ p が，InsertbyIndex ではポインタ r が，Remove ではポインタ q が指すノードが明確に表示すべきノードである．変更した Display 関数は struct Element 型のポインター一つを引数に持ち，各関数（InsertFront, InsertbyIndex, Remove）内で呼び出して使用する．合わせて各関数内で Display 関数を読み出すよう変更せよ．処理前の線形リストの状態に対する操作の出力例として以下のようなものが考えられる．

処理前の線形リストの状態

```
 0: 0x600003830010    100 0x600003830000
 1: 0x600003830000     90 0x60000383c0b0
 2: 0x60000383c0b0     80 0x60000383c0a0
 3: 0x60000383c0a0     70 0x60000383c090
 4: 0x60000383c090     60 0x60000383c080
 5: 0x60000383c080     50 0x60000383c070
 6: 0x60000383c070     40 0x60000383c060
 7: 0x60000383c060     30 0x60000383c050
 8: 0x60000383c050     20 0x60000383c040
 9: 0x60000383c040     10 0x0
```

出力例）InsertbyIndex(5, 110); 後（<- edited node で挿入されたノードを明確にしている）

```
index番目にノードを挿入した後のリスト
     0: 0x600003830010    100 0x600003830000
     1: 0x600003830000     90 0x60000383c0b0
     2: 0x60000383c0b0     80 0x60000383c0a0
     3: 0x60000383c0a0     70 0x60000383c090
     4: 0x60000383c090     60 0x600003834000
     5: 0x600003834000    110 0x60000383c080 <- edited node
     6: 0x60000383c080     50 0x60000383c070
     7: 0x60000383c070     40 0x60000383c060
     8: 0x60000383c060     30 0x60000383c050
     9: 0x60000383c050     20 0x60000383c040
    10: 0x60000383c040     10 0x0
```

問題を解き，`ex12.c` と同様，`InsertFront()`，`Remove()`（先頭ノードの削除，それ以外のノードの削除），`InsertbyIndex()`（先頭への挿入，末尾への挿入，間への挿入） の動作確認を実施せよ．

本課題では指示が少なく実装上悩む場面があるかもしれませんが，要求された仕様がどうしたら実現できるか，線形リストの基本操作の勉強も兼ねて考えてみてください．

課題の提出方法

課題５のレポートについては課題６とまとめて提出することとします．本日のタイミングではレポート提出を課しません．