リスト処理

リストと呼ばれるデータ構造には線形リスト，循環リスト，重連結リストなどがあります．授業では線形リストについて学んでいきます．

線形リスト

順序付けられて並んだデータ構造はリストと呼ばれ，最も単純なものに線形リストと呼ばれるものがあります．線形リストを構成した例を図１に示す．

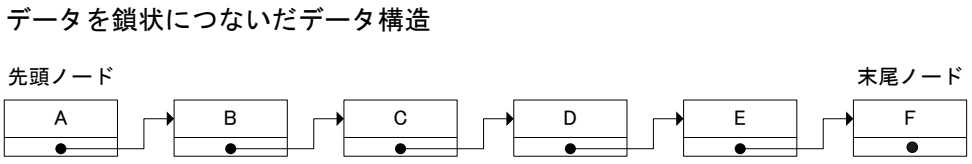

図１　線形リストの一例

リスト上の個々のデータはノードと呼ばれ，各ノードは順番に連結されています．ノードを辿る場合は先頭から順番に辿ることになります．線形リストの最初と最後のノードはそれぞれ先頭ノードおよび末尾ノードと呼ばれ，着目したノードの一つ前のノードは先行ノード，一つ後ろのノードは後続ノードと呼ばれます．ノードはポインタ変数を利用して連結することでリストを構成します．配列と比較すると，データの挿入や削除に伴って，データを移動する処理を行わなくて済みます．また，データの最大数が分からない場合にも構造体オブジェクトを動的に確保することで対応することができます．

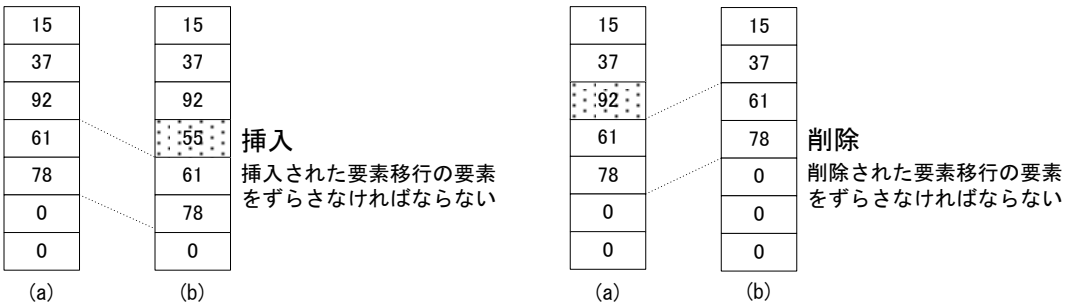

図２　配列のデータの挿入・削除時の問題

線形リストの実現

ノードを表す構造体 Node 型を図３のように定義します．扱いたいデータ本体と自分自身と同じ型を指すポインタをメンバに持ちます．このような構造体は自己参照構造体と呼びます．

```
/*--- ノード ---*/
typedef struct __node {
        Member          data;     /* データ */
        struct __node *next;      /* 後続ポインタ（後続ノードへのポインタ）*/
} Node;
```

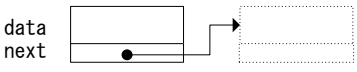

図３　自己参照構造体

後続ノードへのポインタを格納するのがメンバ next です．ただし，後続ノードが存在しない場合は空ポインタ NULL を入れておきます．以下では線形リストを操作するにあたって，処理が複雑になる部分に

ついて解説していきます.

ノードの探索

データ本体に会員番号あるいは氏名を設定したノードを線形探索する手続きを説明します．図４に示すように，目的とする値をもつノードに出会うまで，先頭から順になぞっていくことで実現します.

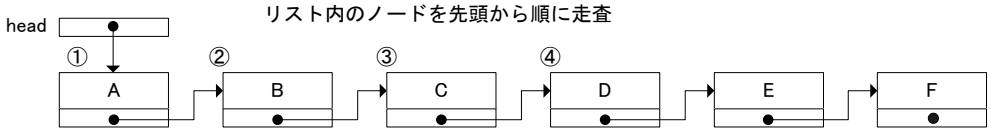

図４　ノードの探索（線形探索）

探索処理における繰り返しの終了条件は以下のいずれかが成立した場合です.

（x1）探索条件を満たすノードが見つからず末端を通り越しそうになった.
（x2）探索条件を満たすノードを見つけた.

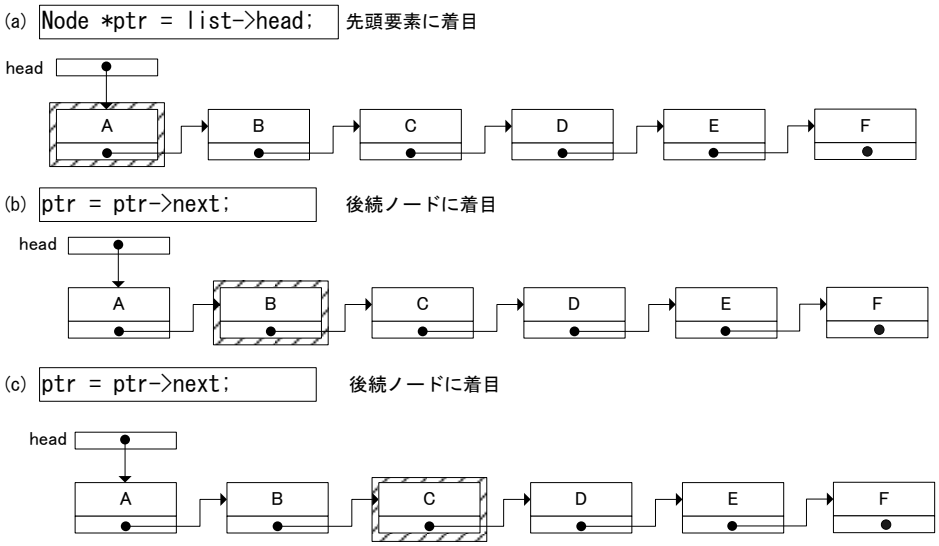

図５　探索のプログラム

先頭へのノードの挿入

リストの先頭にノードを挿入することを考えます．図６（a）に示すリストの先頭にノードＧを挿入すると図６（b）のようになります.

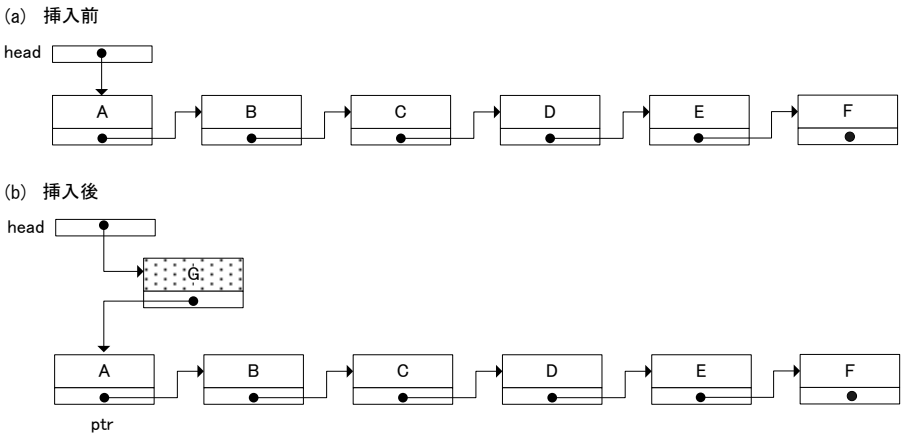

図６　先頭へのノードの挿入

末尾へのノードの挿入

リスト空の場合には先頭にノードを挿入する手続きを行います．リストが空でない場合は，まず，末尾ノードを見つけなければなりません．先頭から空ポインタが見つかるまで順番に辿ります．新しいノードGを連結します．

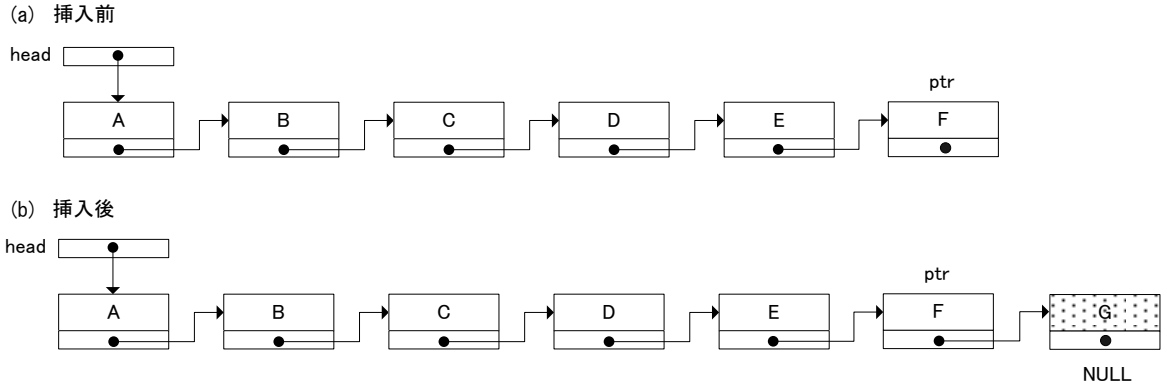

図７　末尾へのノードの挿入

中間へのノードの挿入

例えば，CとDのノードの間に新しいノードを挿入する場合には，Cの後続ポインタに新しいノードGのアドレスを格納し，新しいノードの後続ポインタにDのノードのアドレスを格納することで，中間のノードに挿入することができます．

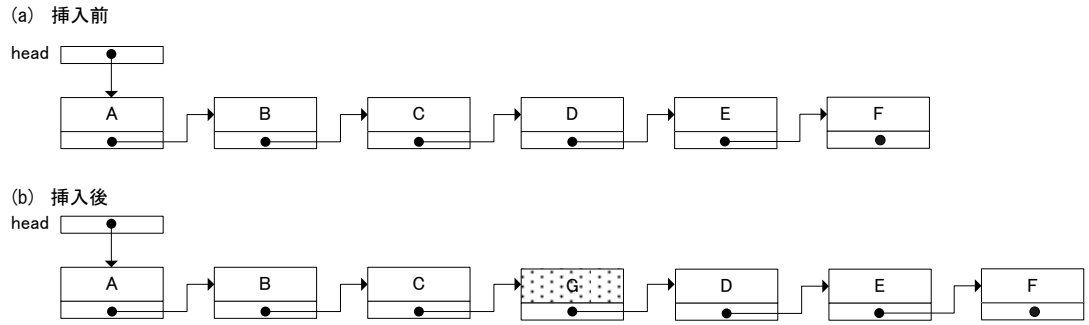

図８　中間へのノードの挿入

先頭ノードの削除

リストが空であれば削除する必要がないので，まずそのチェックを最初に行います．空でない場合には以下の図に従ってノードを削除します．

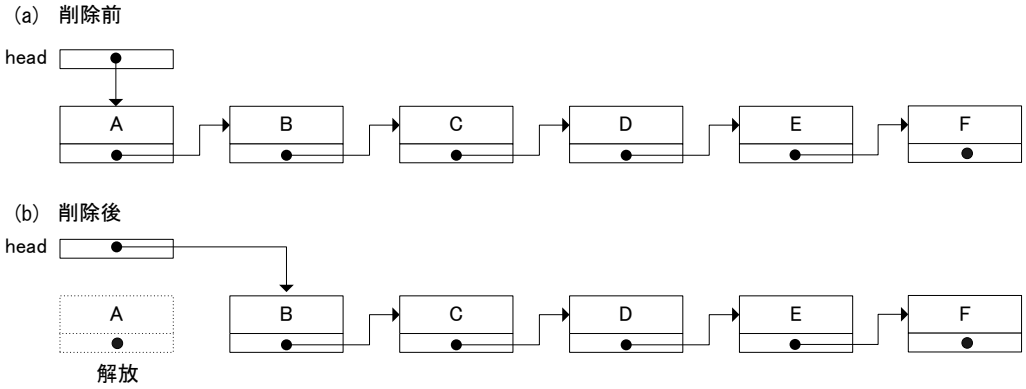

図９　先頭ノードの削除

末尾ノードの削除

リスト上にノードが一つだけ存在する場合には先頭ノードを削除する場合と同じ扱いをします．そうでない場合はまず，末尾から 2 番目のノードを見つけます．そして，ノード E の next を空ポインタにして，ノード F を解放することで末尾ノードを削除できます．

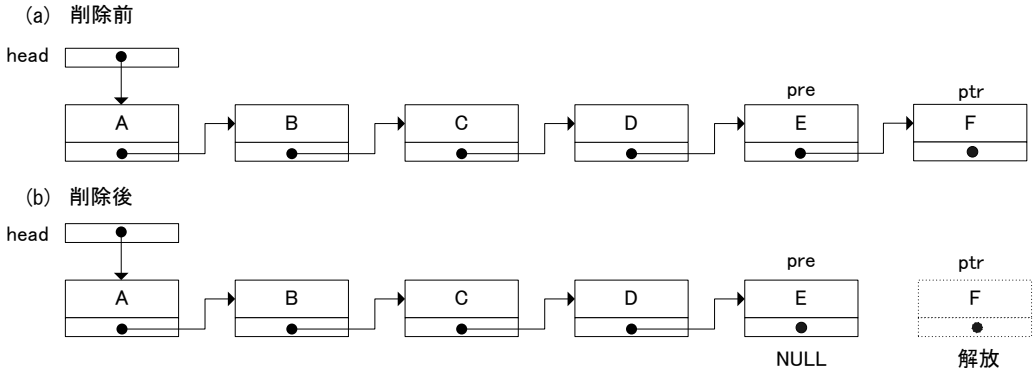

図１０　末尾ノードの削除

着目ノードの削除

着目しているノードが先頭ノードであれば先頭ノードを削除する処理を行います．そうでない場合については以下の図のように処理します．今，削除したい着目ノードをノード D とすると，先行ノード C の next に後続ノード E を設定し，着目ノード D を解放します．

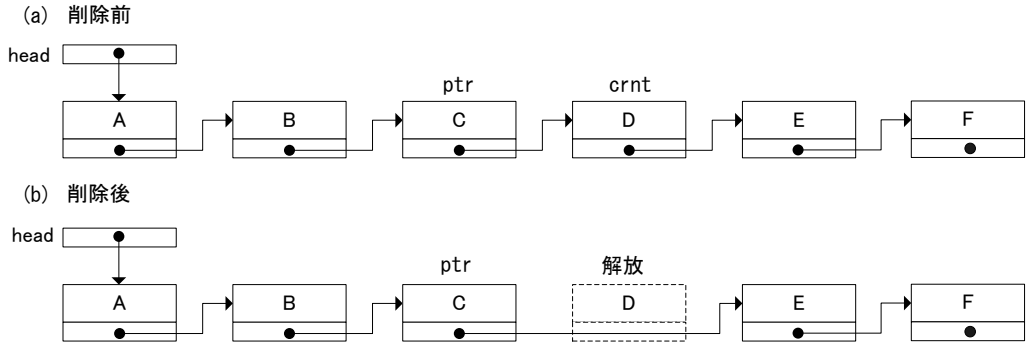

図１１　着目ノードの削除

全ノードの削除

全ノードの削除は線形リストが空になるまで，先頭要素の削除を繰り返すことで行います．

課題５

線形リストを用いたオリジナルのプログラムを作成しなさい．

プログラム例：

(1) ノードには会員番号が昇順に格納されているものとし，追加するノードが適切な位置に追加される関数を追加する．

(2) 線形リストに対するクイックソート（余裕があれば度数ソートも）．

※これらを提出する場合は関数と実行結果で良い．