2019/05/21

2019 年度 アルゴリズム・データ構造演習 (担当：大川剛直)

課題番号：課題 0521，レポート提出期限：2019 年 5 月 27 日 10:40

前回の課題と同様に，以下の構造体 POINT を使って，100 次元空間上の 1 点を表現する．なお，点の第 i 番目 $(0 \leq i \leq 99)$ の軸の座標値 x[i] は，$0 \leq$ x[i]$< M$ を満たす整数値のみを対象とする．

```
typedef struct _point {
  int x[100];     /* 0, 1, 2, ... , M-1 のいずれかの値 */
} POINT;
```

このような点のデータ（へのポインタ）を，2 分ヒープと 2 分探索木を用いて登録することを考える．以下の関数を実装しながら，指示に従い，考察せよ．

- POINT 型の点のデータへのポインタ p を受け取り，2 分ヒープに登録する関数 void insert_heap(POINT *p) を作成せよ．このとき，親節点に登録された点は子節点に登録された点よりも原点に近いか等距離とする．また，空間内で全く同一座標位置の点へのポインタが既に登録済の場合にも，二重登録するものとする．

- POINT 型の点のデータへのポインタ p を受け取り，2 分探索木に登録する関数 void insert_bst(POINT *p) を作成せよ．このとき，ある節点の左部分木には，その節点に登録された点よりも原点に近いか等距離の点データのみを登録し，右部分木には，その節点に登録された点よりも原点から遠い点データのみを登録するものとする．また，空間内で全く同一座標位置の点へのポインタが既に登録済の場合には，二重登録するものとする．

以上のように作成した関数を利用し，以下の手順に従い，考察を行え．

(1) 前回と同様に，関数 malloc を用いて，POINT 型の点データの領域を確保する．0 以上 M 未満の整数の一様乱数を 100 個発生させ，それらを 0 ～ 99 番目の軸の座標値として，確保した領域に代入することにより，1 つの POINT 型の点データを生成する．このとき，各座標が取る値の範囲（M の値）として，$M = 3$ の場合と $M = 100$ の場合を想定すること．

(2) このような点データの生成を N 回繰り返し，生成された点データのアドレスを POINT *型の配列 P[0], P[1], P[2] $\cdots$ P[N-1] に順次格納する[1]．なお，N は定数で大きめの値とする．

(3) 配列 P[0], P[1], P[2] $\cdots$ P[$n-1$] に格納されている n 個の点データのアドレスを，この順に与えながら関数 insert_heap を n 回繰り返す（n 個のデータを登録する）のに必要な時間を計測する．なお，ヒープは空の状態から始めて n 個のデータを登録するものとする．

(4) (3) と同様に，配列 P[0], P[1], P[2] $\cdots$ P[$n-1$] に格納されている n 個の点データのアドレスをこの順に与えながら関数 insert_bst を n 回繰り返す（n 個のデータを登録する）のに必要な時間を計測する．なお，2 分探索木は空の状態から始めて n 個のデータを登録するものとする．

(5) n を変えながら，(3) と (4) を行い，n 個のデータの登録に必要な時間について，2 分ヒープと 2 分探索木で比較せよ．点データを生成する際に，各座標が取る値の範囲（M の値）として，$M = 3$ とした場合と $M = 100$ とした場合で，どのように変化するかについても，プログラムの実行を通して考察せよ．

---

[1] 以下のステップで，$n(\leq$ N) 個のデータをヒープや二分探索木に登録するが，点データを生成しながら登録すると，データを生成するための時間が実行時間に含まれてしまうため，事前に，登録対象となる全ての点データを生成し，そのデータのアドレスを配列に格納しておくこと．また，n を変える度にデータを生成し直すのであれば，N $= n$ としても良いが，一度，生成したデータ（配列）を（n を変えながら）使い回すのであれば，N の値は，n の最大値にすること．