アルゴリズムとデータ構造 第 9 回講義 課題

担当： 茨城大学 機械システム工学科 井上

リスト構造の強力な利点は，記録するデータのサイズに制限がないことである．以下は，これを利用してリストによるスタックを実現するコード例である．ただし，スタックにデータを入出力する関数 push()，pop()の内容は空欄となっている．

なお，ベースとなるソース・コードは講義ホームページおよび Teams 上に置かれている．

(1) これらの関数を作成し，プログラムを完成せよ．

(2) プログラムをコンパイル・実行し，以下の入力を行った際の画面表示を示せ．

入力：「i, 1, d, i, 2, d, o, o, o, d, q」

```
#include <stdio.h>
#include <stdlib.h>

// ノードの構造体
struct tfield {
  int data;
  struct tfield *pt;
};

// ノード型(NODE)の型定義
typedef struct tfield NODE;

NODE *talloc(void);
void disp(NODE *p);
void push(int, NODE **);
int  pop(int *, NODE **);

int main(int argc, char *argv[])
{
  NODE *head = NULL;
  int   n;
  char  command[256];

  while (1) {
    printf("Command: ");
    scanf("%s", command);
    switch(command[0]) {
    case 'i': case 'I':
      printf("Data: ");
      scanf("%d", &n);
      push(n, &head);
      break;
    case 'o': case 'O':
      if (pop(&n, &head) == -1)
        printf("Stack: Empty¥n");
      else
        printf("%d¥n", n);
      break;
    case 'd': case 'D':
      disp(head);
      break;
    }
    if ((command[0] == 'q') ||
        (command[0] == 'Q'))
      break;
    printf("¥n");
  }
}

// ノード新規生成
NODE *talloc(void)
{
  NODE *p;

  p = (NODE *)malloc(sizeof(NODE));
  if (p == NULL) {
    fprintf(stderr, "malloc() failed.¥n");
    exit(EXIT_FAILURE);
  }
  return p;
}

// リスト内容表示
void disp(NODE *p)
{
  printf("Stack: ");
  if (p == NULL)
    printf("Empty¥n");
  else {
    while (p != NULL) {
      printf("%d ", p->data);
      p = p->pt;
    }
    printf("¥n");
  }
}

// データ入力
void push(int n, NODE **head)
{


}

// データ出力
int pop(int *n, NODE **head)
{


}
```

以上