Podstawy programowania

I rok Automatyka i Robotyka Eka PWr

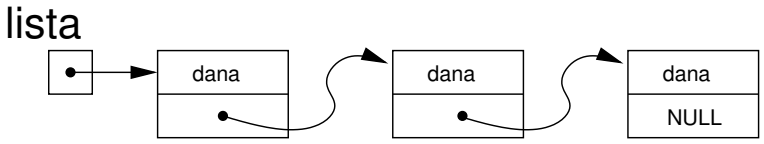
Ćwiczenia – Zestaw 6

Zakres materiału

Listy i stosy.

Zadania

1. Lista jednokierunkowa to dynamiczna struktura danych, której każdy element może pokazywać na kolejny, taki sam element, co zilustrowano na poniższym rysunku.



Dostęp do listy jest możliwy dzięki zmiennej wskaźnikowej pokazującej na jej pierwszy element. Każdy element listy zawiera dwojakiego rodzaju pola: pole (pola) danych, pozwalające na przechowywanie na liście użytecznych informacji oraz pole konstrukcyjne, pozwalające na przechowyanie wskaźnika do następnego elementu listy.

Do przechowywania elementów listy jednokierunkowej w języku C służą struktury o konstrukcji typu

```
typedef struct elem {
  int dana;           /* nasza uzyteczna dana */
  struct elem *nast;  /* pole konstrukcyjne */
} t_elem;
```

Dla tak zadanej struktury napisać zestaw funkcji o podanych poniżej nagłówkach, z których kolejne pozwolą na:

- zainicjowanie pustej listy,
  `t_elem *inicjuj();`
- utworzenie elementu gotowego do dodania do listy,
  `t_elem *tworz(int dana);`
- dodanie elementu na początek listy,
  `t_elem *dodaj_poczatek(t_elem *poczatek, int dana);`
  `t_elem *dodaj_poczatek(t_elem *poczatek, t_elem * elem);`
- dodanie elementu na koniec listy,
  `t_elem *dodaj_koniec(t_elem *poczatek, int dana);`
  `t_elem *dodaj_koniec(t_elem *poczatek, t_elem *elem);`
- dodanie elementu za wskazany element listy,
  `t_elem *dodaj(t_elem *poczatek, t_elem * poprzedni, int dana);`
  `t_elem *dodaj(t_elem *poczatek, t_elem * poprzedni, t_elem *elem);`

- usunięcie elementu z początku listy,
  `t_elem *usun_poczatek(t_elem *poczatek);`
- usunięcie elementu z końca listy,
  `t_elem *usun_koniec(t_elem *poczatek);`
- usunięcie elementu zza wskazanego elementu listy,
  `t_elem *usun_zza(t_elem *poczatek, t_elem *elem);`
- usunięcie wskazanego elementu listy.
  `t_elem *usun(t_elem *poczatek, t_elem *elem);`

Wszystkie funkcje za wyjątkiem funkcji `tworz` zwracają jako swoją wartość wskaźnik na początek listy po jej modyfikacji. Funkcja `tworz` zwraca wskaźnik na utworzony element.

2. Lista dwukierunkowa to dynamiczna struktura danych, różniąca się od listy jednokierunkowej tym, że jej elementy oprócz wskazywania na elementy bezpośrednio po nich następujące, wskazują również na elementy bezpośrednio je poprzedzające.

   Rozbudować strukturę z poprzedniego zadania w taki sposób, by pozwalała na konstruowanie listy dwukierunkowej. Przy tak rozbudowanej strukturze napisać funkcje realizujące na liście dwukierunkowej takie same operacje jak określone w poprzednim zadaniu dla listy jednokierunkowej.

3. Stos, zwany również kolejką LIFO (ang. *last in first out*) jest abstrakcyjną strukturą danych definiowaną poprzez operacje, które można na nim wykonać. Stos to liniowa struktura danych, w której dane dokładane są na jego wierzchołek i z jego wierzchołka są pobierane. Operacje, które można na stosie wykonać to sprawdzenie, czy stos nie jest pusty, złożenie elementu na wierzchołku stosu i pobranie elementu z wierzchołka stosu (jeśli takowy się tam znajduje).

   Zaproponować implementacje stosu, w postaci definicji odpowiedniej struktury danych o nazwie `t_stos` wraz z funkcją zapewniającą jego poprawną inicjację
   ```
   void init(t_stos *stos);
   ```
   i operujących na niej trzech funkcji, odpowiednio
   ```
   int empty(t_stos *stos);
   int push(t_stos *stos, int *dana);
   int pop(t_stos *stos, int *dana);
   ```
   realizujących wymienione powyżej czynności (funkcja `empty` zwraca wartość mówiącą o tym, czy stos jest pusty czy też nie, zaś pozostałe wartość informującą o powodzeniu przeprowadzenia działania), utworzone w oparciu o:

   - listę jednokierunkową,
   - tablicę jednowymiarową.