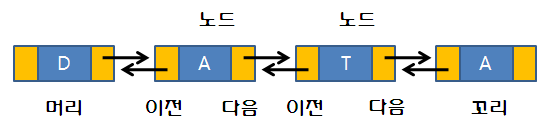
**———————————————————————————————————————————————————**

[1] 기초 문제

———————————————————————————————————————————————————

1. **다음 그림에서 찾을 수 있는 이중 연결 리스트의 용어들에 어떤 것이 있는지 나열하시오.**



1. **이중 연결 리스트 용어와 관련된 문제다. 빈 칸에 들어가는 단어를 맞춰보도록 한다.**
   1. ① 이중 연결 리스트 ② 단순 연결 리스트 ③ 양방향 ④ 두 개 ⑤ 머리 노드와 꼬리 노드
   2. ① 노드 ② 두 개 ③ 다음 노드 ④ 이전 노드
   3. ① 머리 노드 ② 이중 연결 리스트 ③ 단순 연결 리스트 ④ 접근 성능
2. **이중 연결 리스트를 사용하고 있는 곳을 세 가지씩만 나열해 보자.**
   1. 줄 전구, 비엔나 소시지, 강강수월래, 진주 목걸이, 징검다리
   2. 파워포인트 객체 관리, CPU 스케줄링, 인터넷 만화, 메신저
3. **이중 연결 리스트는 배열보다 단순 연결 리스트와 경쟁을 할 수밖에 없다. 빈 칸에 들어가는 내용을 단답형으로 채워보자.**

|  |  |  |
| --- | --- | --- |
| 구분 | 단순 연결 리스트 | 이중 연결 리스트 |
| 포인터 개수 | 1개(next) | 2개(prev, next) |
| 진행 방향 | 순방향(오른쪽으로만 이동) | 양방향(왼쪽과 오른쪽으로 모두 이동) |
| 직접 접근 노드 | 머리 노드 | 머리 노드, 꼬리 노드 |
| 역방향 순회 방법 | 재귀 함수 사용 | 반복문 사용 |

|  |
| --- |
| 해답  표에 대한 완전한 설명은 4장의 연계 학습에 있습니다. 설명이 너무 길어 기본적인 답안만 제공합니다. |

1. **본문에 나왔던 코드에 대해 설명을 하는 곳이다. 이중 연결 리스트를 배우려는 친구에게 설명한다는 생각으로 차분하게 기술해 보자.**

|  |
| --- |
| 해답  자세한 설명이 2장의 심화 학습에 있습니다. 설명이 너무 길어 답안은 생략했습니다. |

**———————————————————————————————————————————————————**

[2] 기본 실습 문제

———————————————————————————————————————————————————

1. **본문에서 배웠던 가벼운 함수들이다. 직접 구현해 보고 본문 코드와 비교해 보도록 하자.**
   1. NODE\* makeNode(char data)

{

NODE\* node = malloc(sizeof(NODE));

node->data = data;

node->prev = node->next = NULL;

return node;

}

* 1. char getTail(DOUBLE\_LIST\* pList)

{

return pList->pTail->data;

}

1. **본문에서 배웠던 조금 무거운 함수들이다. 직접 구현해 보고 본문 코드와 비교해 보도록 하자.**
   1. void printReverse(DOUBLE\_LIST\* pList)

{

NODE\* cur;

for(cur = pList->pTail; cur != NULL; cur = cur->prev)

printf("%c ", cur->data);

printf("\n");

}

* 1. void addTail(DOUBLE\_LIST\* pList, char data)

{

NODE\* node = makeNode(data);

if(isEmpty(pList) == 1)

{

pList->pHead = pList->pTail = node;

}

else

{

node->prev = pList->pTail;

node->next = NULL;

pList->pTail->next = node;

pList->pTail = node;

}

}

1. **저자는 여러분이 복잡한 코드까지 직접 구현하기를 바라지는 않는다. 대신 비어있는 곳의 코드를 채울 정도는 되어야 한다고 생각한다. 빈 칸에 들어가는 코드는 무엇인가?**
   1. ① if(pList->pHead == pList->pTail)

② pList->pHead = pList->pTail = NULL;

③ free(del);

* 1. ① if(isEmpty(pList) == 1)

② pList->pHead->prev = node;

③ pList->pHead = node;

**———————————————————————————————————————————————————**

[3] 응용 실습 문제

———————————————————————————————————————————————————

1. char medianNode(DOUBLE\_LIST\* pList)

{

NODE\* left = pList->pHead,

\* right = pList->pTail;

if(isEmpty(pList) == 1)

return '\0';

while(1)

{

if(left == right)

break;

left = left->next;

if(left == right)

break;

right = right->prev;

}

return left->data;

}

1. 구현이 어려워서 테스트 코드까지 함께 실었습니다.

#include <stdio.h>

#include <ctype.h>

#include "DoublyLinked.h"

void moveAlpha(DOUBLE\_LIST\* src, DOUBLE\_LIST\* upper, DOUBLE\_LIST\* lower)

{

NODE\* cur;

upper->pHead = upper->pTail =

lower->pHead = lower->pTail = NULL;

for(cur = src->pHead; cur; cur = cur->next)

{

if(isalpha(cur->data) != 0)

{

// 이전과 다음 노드 연결 : cur 분리

if(cur->prev) cur->prev->next = cur->next;

else src->pHead = cur->next;

if(cur->next) cur->next->prev = cur->prev;

else src->pTail = cur->prev;

// 목표 리스트에 연결

addTail(isupper(cur->data) ? upper : lower, cur->data);

}

}

}

void main(void)

{

DOUBLE\_LIST list = { 0 }, t1, t2;

char s1[] = "tsR789BLACK123black45AbcdE";

int i;

for(i = 0; s1[i]; i++)

addTail(&list, s1[i]);

printf("정상: "); printNormal(&list);

printf("반대: "); printReverse(&list);

moveAlpha(&list, &t1, &t2);

printf("\n");

printf("정상: "); printNormal(&list);

printf("반대: "); printReverse(&list);

printf("\n");

printNormal(&t1);

printNormal(&t2);

}