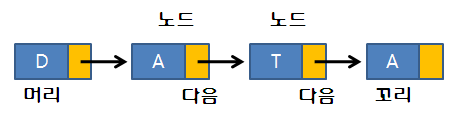
**———————————————————————————————————————————————————**

[1] 기초 문제

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

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



1. **리스트 용어와 관련된 문제다. 빈 칸에 들어가는 단어를 맞춰보도록 한다.**
   1. ① 연결 리스트 ② 배열 ③ 노드 ④ 포인터 멤버 ⑤ 순회
   2. ① 노드 ② 요소 ③ 데이터 ④ 다음 ⑤ 구조체
   3. ① 머리 노드 ② 첫 번째, 가장 앞쪽, 최초 ③ 접근 성능
2. **리스트를 사용하고 있는 곳을 세 가지씩만 나열해 보자.**
   1. 끝말 잇기, 연등, 비밀 결사, 여행 일정표, 전기 게임
   2. 링크, 문서 편집, 파일 시스템, MFC에서의 도큐먼트 템플릿, 리눅스 i-node
3. **배열과 리스트는 상호 보완적인 관계로 두 가지 자료 구조만 있으면 못할 것이 없다. 빈 칸에 들어가는 내용을 단답형으로 채워보자.**

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 세부 항목 | 배열 | 리스트 |
| 메모리 | 연속 | 예 | 아니오 |
| 할당 | 충분한 크기로 미리 할당 | 데이터를 추가할 때마다 할당 |
| 해제 | 프로그램 종료시에 일괄 해제 | 데이터를 삭제할 때마다 해제 |
| 사용량 | 적다(데이터만) | 많다(데이터+포인터) |
| 연산 | 순회 | 좋다 | 나쁘다 |
| 탐색 | 빠르다 | 느리다 |
| 삽입 | 최악 | 최고 |
| 삭제 | 최악 | 최고 |

|  |
| --- |
| 해답  표에 대한 완전한 설명은 2장의 심화 학습에 있습니다. 설명이 너무 길어 기본적인 답안만 제공합니다. |

1. **본문에 나왔던 코드에 대해 설명을 하는 곳이다. 단순 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. char getHead(NODE\* pHead)

{

return pHead->next->data;

}

* 1. void addTail(NODE\* pHead, char data)

{

NODE\* cur = pHead;

while(cur->next != NULL)

cur = cur->next;

addHead(cur, data);

}

* 1. int isEmpty(NODE\* pHead)

{

return pHead->next == NULL;

}

1. **본문에서 배웠던 조금 무거운 함수들이다. 직접 구현해 보고 본문 코드와 비교해 보도록 하자.**
   1. char getTail(NODE\* pHead)

{

NODE\* cur = pHead;

while(cur->next != NULL)

cur = cur->next;

return cur->data;

}

* 1. int countNode(NODE\* pHead)

{

int count;

NODE\* cur = pHead->next;

for(count = 0; cur != NULL; count++)

cur = cur->next;

return count;

}

* 1. void clear(NODE\* pHead)

{

NODE\* cur, \* next = pHead->next;

while(cur = next)

{

next = cur->next;

free(cur);

}

pHead->next = NULL;

}

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 NODE\* cur = pHead;

② cur->next != NULL

③ cur = cur->next;

* 1. ① if(cur->data == data)

② return cur;

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

[3] 응용 실습 문제

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

1. char getData(NODE\* pHead, int index)

{

int i;

NODE\* cur;

i = 0, cur = pHead->next;

while(i < index && cur)

i++, cur = cur->next;

return cur ? cur->data : '\0';

}

1. void printReverse(NODE\* pNode)

{

if(pNode != NULL)

{

printReverse(pNode->next);

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

}

}

1. void reverseList(NODE\* pHead)

{

NODE\* prev, \* cur, \* next;

next = pHead->next;

cur = NULL;

while(next)

{

prev = cur;

cur = next;

next = next->next;

cur->next = prev;

}

pHead->next = cur;

}