알고리즘(Algorithm)

작성자: 박용준 강사(http://www.VisualAcademy.com)

1 알고리즘(Algorithm)과 절차 지향 프로그래밍

이번 강좌에서는 배우기 쉬운 알고리즘을 몇 가지 소개합니다. 지금까지 배운 프로그래밍 언어의 기능으로 특정 문제를 해결하는 코드를 작성하는 기법을 배우겠습니다. 알고리즘을 바탕으로 입력, 처리, 출력의 단계로 진행되는 프로그래밍 언어의 절차 지향 프로그래밍 기법을 정리합니다.

1.1 알고리즘(Algorithm)

알고리즘(풀이법)이란 프로그램 개발에 있어서 필요한 문제를 해결하는 방법을 체계적으로 정리하는 방법이라 볼 수 있습니다. 주어진 문제를 어떻게 풀이하는가에 따라서 그 문제를 해결할 수도 있고 그렇지 못할 수도 있는 것입니다. 이 때문에 프로그램 작성에 있어 알고리즘이란 중요한 자리를 차지하고 있는 것입니다.

- 알고리즘은 "문제 해결 능력"입니다.
- 프로그램의 가장 작은 단위는 일반적으로 입력(Input) -> 처리(Process) -> 출력(Output)의 단계를 거치는데, 여기서 처리의 단계가 알고리즘의 단계로 보면 됩니다.
  - 입력: 자료 구조(Data Structure)가 담당하는 영역입니다.
    - 간단히는 변수 및 배열의 데이터가 사용되고 더 나아가서는 컬렉션, 파일, 데이터베이스의 데이터가 사용되는 영역입니다.
  - 처리: 알고리즘 처리 영역입니다.
  - 출력: UI가 담당하는 영역입니다. 일반적으로 콘솔, 데스크톱, 웹, 모바일 등의 영역으로 나누어서 가공된 데이터가 출력됩니다.

1.1.1 순서도(FlowChart)

알고리즘을 정해진 기호로 표시한 것을 순서도라고 합니다. 단, 이 강의에서는 따로 순서도는 다루지 않습니다. 만약, 응용 프로그램 제작이 아닌 정보처리기사와 같은 자격증 취득이 목적이라면 순서도가 그때에는 필요할 수도 있습니다.

1.2 학습용 쉬운 알고리즘

가장 처음에 배워야하는 학습용 알고리즘은 주어진 자료를 가지고 가장 큰 값을 구하던가(최댓값), 가장 작은 값을 구하던가(최솟값), 합을 구하던가(누적합), 자료수를 구하던가(횟수, 건수), 평균을 구하던가(평균), 순서대로 정렬하던가(정렬) 등이 있습니다. 다음 표는 필자가 생각하는 알고리즘 입문용으로 가장 적합한 내용을 정리해 보았습니다.

난이도	알고리즘	사용 유형
초급	합계(SUM)	합계를 출력하시오.
	개수(COUNT; 횟수, 건수)	자료 건수를 출력하시오.
	평균(AVERAGE)	평균을 출력하시오.
	최댓값(MAX)	최댓값을 출력하시오.
	최솟값(MIN)	최솟값을 출력하시오.
중급	최댓값(MAX) → 최솟값(MIN)	~에 대해서 최댓값을 구하되, 동일값 발생시 ~에 대해서 최솟값을 구하시오.
	최솟값(MIN) → 최댓값(MAX)	~에 대해서 최솟값을 구하되, 동일값 발생시 ~에 대해서 최댓값을 구하시오.
	최댓값(MAX) → 최댓값(MAX)	~에 대해서 최댓값을 구하되, 동일값 발생시 ~에 대해서 최댓값을 구하시오.
	최솟값(MIN) → 최솟값(MIN)	~에 대해서 최솟값을 구하되, 동일값 발생시 ~에 대해서 최솟값을 구하시오.
	최댓값(MAX) - 최솟값(MIN)	~에 대해서 최댓값과 최솟값을 구하고, 최댓값과 최솟값을 차를 구하시오.
고급	근삿값(NEAR)	~에 가장 가까운 값을 구하시오.

	순위(RANK)	순위를 구하시오.
	정렬(SORT): 오름차순	~에 대해서 오름차순 정렬하시오.
	정렬(SORT): 내림차순	~에 대해서 내림차순 정렬하시오.
	검색(SEARCH)	특정 자료를 검색하시오.
	병합(MERGE)	2개의 배열을 하나의 배열로 합치시오.
	최빈값(MODE)	가장 빈도수가 높은 자료를 구하시오.
	그룹(GROUP)	특정 항목별 그룹화하여 소계를 구하시오.

앞으로 12개의 쉬운 알고리즘을 학습할텐데요. 모든 소스 코드는 디버거를 사용하여 줄 단위로 코드를 실행하면서 코드의 흐름을 익히는 것을 권장합니다.

1.3 정렬(SORT) 알고리즘

1.3.1 정렬(SORT) 알고리즘

주어진 범위내에서 불규칙적으로 나열된 순서를 일정 기준에 따라 순서대로 나열하는 알고리즘입니다. 정렬 알고리즘은 선택 정렬(Selection Sort), 버블 정렬(Bubble Sort), 퀵 정렬(Quick Sort) 등 많이 있는데 이 강의에서는 선택 정렬 알고리즘만 사용할 것입니다.

- 오름차순(ASCeding) 정렬은 1, 2, 3 순으로 작은 것부터 큰 순으로 정렬합니다.
  - 1, 2, 3
  - 가, 나, 다
  - A, B, C
- 내림차순(DESCending) 정렬은 3, 2, 1 순으로 큰 것부터 작은 순으로 정렬합니다.
  - 3, 2, 1
  - ㄷ, ㄴ, ㄱ
  - c, b, a

1.3.2 선택정렬(SELECTION SORT) 알고리즘

데이터 하나를 기준으로 나머지 데이터와 비교하여 가장 작거나 큰 데이터와 자리를 바꾸는 식으

로 반복 비교하는 정렬 방법입니다. 선택 정렬은 데이터의 개수가 n개이면 전체 회전수는 n-1회입니다. 선택 정렬 알고리즘은 오름차순 기준으로 배열의 처음부터 가장 작은 데이터가 채워집니다.

1.3.3 선택정렬 회전수

배열 data[5]에 다음과 같이 데이터가 입력되어 있다고 할 때 선택 정렬을 사용해서 오름차순으로 정렬시키는 단계를 간단히 표현해 보겠습니다. 지면상 모든 단계를 표현하는게 아닌 왼쪽에 가장 작은 값이 들었을 때까지만 표현하겠습니다.

data[5]

data[0]	data[1]	data[2]	data[3]	data[4]
46	32	11	24	55

1회전:

data[0]을 기준으로 나머지 데이터와 비교하여 가장 작은 값과 자리를 바꾸는 과정을 반복하면 data[0]에는 가장 작은 값이 들어갑니다.

46	32	11	24	55
32	46	11	24	55
11	46	32	24	55

2회전:

data[1]을 기준으로 나머지 데이터와 비교하여 가장 작은 값과 자리를 바꾸는 과정을 반복합니다. 2회전이 끝나면 data[1]에 두 번째로 작은 값이 들어갑니다.

11	**46**	32	24	55
11	**32**	46	24	55
11	**24**	46	32	55

3회전:

data[2]을 기준으로 나머지 데이터와 비교하여 가장 작은 값과 자리를 바꾸는 과정을 반복합니다. 3회전이 끝나면 data[2]에 세 번째로 작은 값이 들어갑니다.

11	**24**	**46**	32	55
11	**24**	**32**	46	55

4회전:

data[3]을 기준으로 나머지 데이터와 비교하여 가장 작은 값과 자리를 바꾸는 과정을 반복합니다. 4회전이 끝나면 data[3]에 네 번째로 작은 값이 들어갑니다.

11	**24**	**32**	**46**	55
11	**24**	**32**	**46**	55

1.3.3.1 참고: 선택 정렬 관련 정보처리기사 필기 문제

선택 정렬 알고리즘의 흐름을 조금 더 정리하는 차원에서, 정보처리기사 필기 시험에 여러 해에 걸쳐서 출제된 선택 정렬 관련 문제를 참고용으로 풀어보겠습니다.

문제: 자료가 다음과 같이 주어졌을 때 선택 정렬(selection sort)을 적용하여 오름차순으로 정렬할 경우 pass 2를 진행한 후의 정렬된 값으로 옳은 것은?

자료: 9, 4, 5, 11, 8

가. 4, 5, 9, 8, 11　　나. 4, 5, 9, 11, 8

다. 4, 5, 8, 11, 9　　라. 4, 5, 8, 9, 11

답: 나

해설: 가장 작은 데이터를 왼쪽으로 하나씩 채우는 형태로 각 회전이 끝난 후의 배열 모양은 다음과 같습니다.

1. pass 1: 4, 9, 5, 11, 8
2. pass 2: 4, 5, 9, 11, 8
3. pass 3: 4, 5, 8, 11, 9

4. pass 4: 4, 5, 8, 9, 11

1.4 검색(SEARCH) 알고리즘

1.4.1 검색(SEARCH) 알고리즘

검색 알고리즘은 배열 등의 데이터에서 특정 값을 검색하는 알고리즘입니다. 일반적으로 순차 검색과 이진 검색 등으로 구분할 수 있습니다.

- 순차 검색: 전체 데이터를 처음부터 끝까지 순서대로 검색합니다.
- 이진 검색: 정렬되어 있는 데이터를 절반으로 나눠서 검색합니다.

1.4.2 이진 검색(Binary Search) 알고리즘

이진 검색 알고리즘은 주어진 데이터가 오름차순으로 정렬되어 있다고 가정합니다. 만약, 실제 데이터가 정렬되어 있지 않다면, 앞서 배운 정렬 알고리즘 등을 이용하여 정렬한 다음에 이진 검색 알고리즘 로직을 적용해야 합니다.
이진 검색 알고리즘은 영어로 Divide and Conquer(나누기 및 정복) 알고리즘으로 표현하는데 그 의미 그대로 데이터를 절반으로 나눠서 검색하여 순차 검색보다 효율을 높입니다.

다음 그림과 같이 1 차원 배열에 1, 3, 5, 7, 9 의 데이터가 있을 경우 이진 검색을 사용하면 중간 인덱스 값을 찾는 것이 핵심 로직입니다.
중간 인덱스를 mid 로 놓고 low 인덱스는 0 그리고 high 인덱스는 4 로 본 후 각 회전마다 중간 인덱스를 구하고 중간 인덱스의 값이 찾으려는 데이터인지를 비교하면 됩니다.

	0	1	2	3	4	
	1	3	5	7	9	
1회전	1	3	5	7	9	mid = (0 + 4) / 2 = 2
2회전				7	9	mid = (3 + 4) / 2 = 3
3회전					9	mid = (4 + 4) / 2 = 4

위 그림과 아래 설명을 참고해서 한 번 읽어보고 넘어갑니다.

1. 1 회전:
   A. 1 회전 들어가기 전: low = 0, high = 4, mid = (low + high) / 2 = (0 + 4) / 2 = 2
   B. 1 회전: mid 값인 2 인덱스의 데이터인 5 와 찾으려는 9 비교, 찾으려는 데이터가 5 보다 크므로 왼쪽 영역은 버리고 오른쪽 영역만 비교하기 위해서 low 값을 mid + 1 로 증가해서 low 를 3 으로 재설정
2. 2 회전:
   A. 2 회전 들어가기 전: low = 3, high = 4, mid = (3 + 4) / 2 = 3
   B. 2 회전: mid 값인 3 인덱스의 데이터인 7 과 찾으려는 9 비교, 찾으려는 데이터가 7 보다 크므로 왼쪽 영역은 버리고 오른쪽 영역만 비교하기 위해서 low 값을 mid + 1 로 증가해서 low 를 4 로 재설정
3. 3 회전:
   A. 3 회전 들어가기 전: low = 4, high = 4, mid = (4 + 4) / 2 = 4
   B. 3 회전: mid 값인 4 인덱스의 데이터인 9 와 찾으려는 9 비교, 3 번의 검색 끝에 데이터를 찾음

1.4.3 참고: 이진 검색 관련 정보처리기사 필기 문제

이진 검색 알고리즘을 사용하여 데이터를 검색하는 내용 관련해서 코드를 만들기 전에 이해를 돕기 위한 이론적인 문제를 먼저 풀어보겠습니다.

문제: 다음과 같이 레코드가 구성되어 있을 때, 이진 검색 방법으로 14 를 찾을 경우 비교되는 횟수는?

"1 2 3 4 5 6 7 8 9 10 11 12 13 14 15"

가. 2 번 나. 3 번 다. 4 번 라. 5 번

답: 나

해설:

1. 1 회전: (0 + 14) / 2 = 7 번째 인덱스의 값인 8
2. 2 회전: (8 + 14) / 2 = 11 번째 인덱스의 값인 12
3. 3 회전: (12 + 14) / 2 = 13 번째 인덱스의 값인 14 <- 찾으려는 값