RV32I - Bubble Sort 분석

1. C Code분석

전체 코드

```c
void sort(int *pData, int size);
void swap(int *pA, int *pB);

int main()
{
    int arData[6] = {5,4,3,2,1};

    sort(arData, 5);

    return 0;
}

void sort(int *pData, int size)
{
    for (int i =0; i< size; i++){
        for(int j = 0; j<size-i-1;j++){
            if(pData[j] > pData[j+1])
                swap(&pData[j],&pData[j+1]);
        }
    }
}

void swap(int *pA, int *pB)
{
    int temp;
    temp = *pA;
    *pA = *pB;
    *pB = temp;
}
```

Main 함수

```c
int main()
{
    int arData[6] = {5,4,3,2,1};

    sort(arData, 5);

    return 0;
}
```

- 사이즈 6의 배열 정의
- 5부터 1까지 역순으로 저장
- `sort`함수를 거치면 크기순으로 정렬됨

sort 함수

```
void sort(int *pData, int size)
{
    for (int i =0; i< size; i++){
        for(int j = 0; j<size-i-1;j++){
            if(pData[j] > pData[j+1])
                swap(&pData[j],&pData[j+1]);
        }
    }
}
```

- 배열의 시작주소(== 배열의 이름)와 배열의 크기를 입력으로 받음
- j에 해당하는 index와 그보다 큰 index(다음 순서의 값)의 값을 비교하고, 이전의 값이 더 크면 다음 인덱스의 값과 값을 switching해줌
- 한번 수행할 때마다 j의 마지막 인덱스에 비교대상들 중 가장 큰 값이 배정됨
- 마지막 인덱스를 줄이며 이를 반복하면 크기순으로 정렬됨

swap 함수

```
void swap(int *pA, int *pB)
{
    int temp;
    temp = *pA;
    *pA = *pB;
    *pB = temp;
}
```

- 해당 인덱스를 가리키는 배열의 주소를 입력으로 받음
- 두 포인터의 위치를 바꿈
    - 포인터가 가리키는 주소의 값의 위치가 바뀜
    - 결과적으로 인덱스에 해당하는 값이 바뀜