객체지향프로그래밍 실습과제보고서 #7

2019 년 4 월 26 일

김예원
소프트웨어학과

실습과제 #6: vector 클래스 사용해보기

: 크기 변형이 자유로운 vector 클래스를 이용해 단어의 빈도수를 확인하는 프로그램 작성

1. 단어와 단어의 빈도수를 저장할 WORD 구조체를 정의한다.

***소스코드**

```
struct WORD {
	string str;
	int count;
};
```

단어를 저장하는 str과 빈도수를 나타내는 count를 지정했다.

2. 여러 개의 단어와 각 단어의 사용 빈도수를 보관하기 위해서 vector 클래스를 사용한다.

***소스코드**

```
vector<WORD*> words;
```

words라는 이름의 vector 클래스를 정의했다. WORD 구조체를 가리키고 있는 포인터를 저장하는 배열이다.

3. 키보드에서 문자열을 연속적으로 입력받고, -1이 입력되면 입력을 종료시킨다.

***소스코드**

```
	string buffer;
	while (cin >> buffer) { //단어가 입력되면 CountWords 함수에 넘긴다
		CountWords(buffer);

		if (buffer == "-1") { //-1이 넘어오면 while문 종료
			break;
		}
	}
```

```
void CountWords(string buffer) {
	if (buffer == "-1") {
		return; //-1이 넘어오면 함수를 종료시킨다
	}
	(중략)
}
```

buffer에 단어가 입력되면 CountWords 함수로 넘어가게 했다. 만약 buffer에 담긴 문자열이 -1일 경우, 배열에 입력되는 것이 아니라, 함수를 종료시켜 main의 다음 함수들이 동작하도록 했다.

4. 단어를 새로 입력하고 빈도를 세는 함수 CountWords()

***소스코드**

```
void CountWords(string buffer) {
	if (buffer == "-1") {
		return; //-1이 넘어오면 함수를 종료시킨다
	}

	for (int i = 0; i < words.size(); i++) {
		if (buffer == words[i]->str) {
			words[i]->count++;
			return;
		}
	} //단어가 발견되면 빈도수를 증가시킨다

	WORD *pWord = new WORD;  //발견되지 않으면 새롭게 저장하고 빈도를 1로 지정한다
	pWord->str = buffer;
	pWord->count = 1;

	words.push_back(pWord);    //vector에 pWord의 주소를 삽입

	return;
}
```

buffer를 통해 문자열이 넘어오면 먼저 -1인지 확인한다. 그 다음 기존의 배열이 가리키는 구조체의 단어와 일치한지 검색한다. 일치하면 count 수를 증가시킨다. 발견되지 않으면 동적할당을 통해 새로운 struct를 가리키도록 한다.

5. 단어와 빈도를 출력하는 함수 PrintWords()

***소스코드**

```
void PrintWords() //단어와 빈도를 출력하는 함수
{
	cout << "=========" << endl;
	for (int i = 0; i < words.size(); i++) {
		cout << words[i]->str << " : " << words[i]->count << endl;
	} //vector의 크기만큼 출력한다
	cout << "=========" << endl;

	return;
}
```

words.size()를 이용해 배열의 크기만큼 반복문을 돌려 단어와 빈도수를 출력한다.

6. 동적할당을 해제하는 RemoveAll()

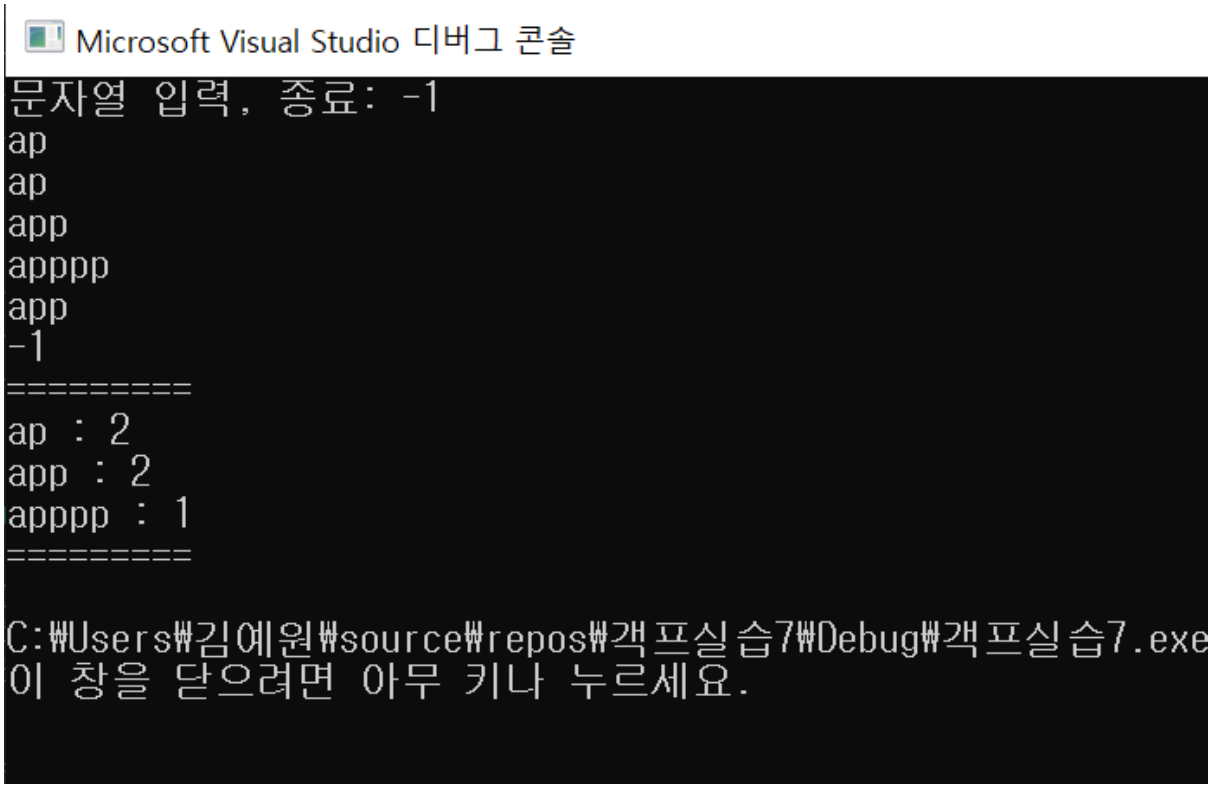
***소스코드**

```
void RemoveAll()
{
	/*for (vector<WORD*>::iterator it=words.begin(); it!=words.end();
it++) {
		delete(*it);
	}
	words.clear();*/

	for (int i = 0; i < words.size(); i++) { //동적메모리를
할당해제한다
		delete words[i];
	}
	words.clear();
}
```

포인터배열이 가리키고 있던 동적메모리를 하나씩 할당 해제했고 words.clear()을 통해 words 내부를 비웠다. 주석 처리된 부분은 vector의 멤버함수 begin과 end를 이용해 할당해제하는 방법이다. 여러가지 방법을 도전해봤다.

이번 과제 실행화면



-1이 입력되면 문자열 입력을 멈추고 최종적인 문자열과 빈도수를 출력한다.

느낀점

중간고사를 본 뒤로 여러가지 생각이 많아졌었다. 많이 공부하고 준비했다고 생각했었는데, 중간고사 문제를 풀면서 느꼈던 것은, 사소한 부분들을 놓치는 공부를 하고 있었다는 것이었다. 과제를 해결하다가 모르거나 오류가 뜨면 내 힘으로 해결하려는 시도를 했다가, 조금 안되면 그냥 주변에 물어봤던 것 같다. 그런 습관을 고쳐야겠다고 생각했다. 이번에 vector에 대한 문제를 해결하면서도 오류가 났었다. 아무리 봐도 알고리즘에는 문제가 없는데 디버깅에러가 자꾸 났다. 혼자 힘으로 해결하고 싶어서 유튜브에 vector 클래스에 대한 인강을 찾아보기도 했고 각종 블로그를 읽어보면서 vector에 대한 공부를 했다. 그러다가 강의자료를 다시 한번 꼼꼼히 읽는 시간을 가졌고, vector.size()라는 멤버변수를 발견했다. 그래서 반복문 안에 sizeof(words)를 지우고 words.size()를 입력했더니, 오류가 해결된 것을 발견했다. 그렇게 sizeof()의 의미에 대해 다시 생각해보게 되었는데, sizeof()는 바이트 수를 계산해주는 함수임을 다시 상기시키게 되었다. 아무것도 아닌 것처럼 보였으나 꽤 큰 실수였다. 이런 식으로 vector 동적할당 해제에 대해서도 공부했다. 처음에 반복문 안에 delete[] words;라고 적었었는데 오류가 떴다. 구조체 포인터를 가리키는 배열이라서 []를 사용했는데, 왜 이런 오류가 뜰까, 궁금했다. 찾아보니까 STL에서는 원소 하나하나 해제해줘야 하기 때문임을 알 수 있었다. 또한 vector 클래스에 대해 배우고 있으니, vector 멤버함수를 이용해서 동적할당 해제하는 법을 알고 싶었다. 강의 자료에 있는 begin과 end를 이용하면 될 것 같았는데, 마음먹은 대로 되지 않았다. 알고보니 인덱스를 참조하는 것이기 때문에 iterator을 이용했어야 했다.

과제를 스스로 공부하면서 해결하는 것은 시간이 많이 필요한 일이었다. 그래도 그만큼 배울 수 있는 것들이 많았고, 스스로 해냈다는 뿌듯함과 새로운 것을 알아냈다는 즐거움이 있어서 행복했다.