REPORT



|  |  |
| --- | --- |
| 과 목 : | 객체지향프로그래밍 |
| 제출일자 : | 2023.04.11 |
| 담당교수 : | 황성호 |
| 학 과 : | 멀티디자인학과 |
| 학 번 : | 201522405 |
| 이 름 : | 최준하 |

1. (예제 4-8) circle 객체의 동적 생성 및 반환 응용.

#include <iostream>

using namespace std;

class \_circle {

int radius;

public:

\_circle();

\_circle(int r);

~\_circle();

void set\_radius(int r) { radius = r; }

double get\_area() { return radius \* radius \* 3.14; }

};

\_circle::\_circle() {

radius = 1;

cout << "생성자 실행 radius = " << radius << endl;

}

\_circle::\_circle(int r) {

radius = r;

cout << "생성자 실행 radius = " << radius << endl;

}

\_circle::~\_circle() {

cout << "소멸자 실행 radius = " << radius << endl;

}

int main() {

int radius;

while (true)

{

cout << "정수 반지름 입력(음수이면 종료)>> ";

cin >> radius;

if (radius < 0) break;

\_circle\* p = new \_circle(radius);

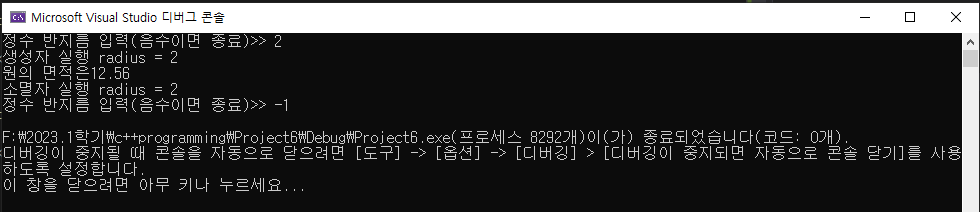
cout << "원의 면적은" << p->get\_area() << endl;

delete p;

}

}

🡪출력 결과



2. (예제 4-10) circle 배열의 동적 생성 및 반환 응용.

#include <iostream>

using namespace std;

class \_circle {

int radius;

public:

\_circle();

~\_circle() {};

void set\_radius(int r) { radius = r; }

double get\_area() { return 3.14 \* radius \* radius; }

};

\_circle::\_circle() {

radius = 1;

}

int main() {

cout << "생성하고자 하는 원의 개수? ";

int n, radius;

cin >> n;

if (n <= 0) {

return 0;

}

\_circle\* p\_array = new \_circle[n];

for (int i = 0; i < n; i++)

{

cout << "원" << i + 1 << ": ";

cin >> radius;

p\_array[i].set\_radius(radius);

}

int count = 0;

\_circle\* p = p\_array;

for (int i = 0; i < n; i++)

{

cout << p->get\_area() << ' ';

if (p->get\_area() >= 100 && p->get\_area() <= 200) {

count++;

}

p++;

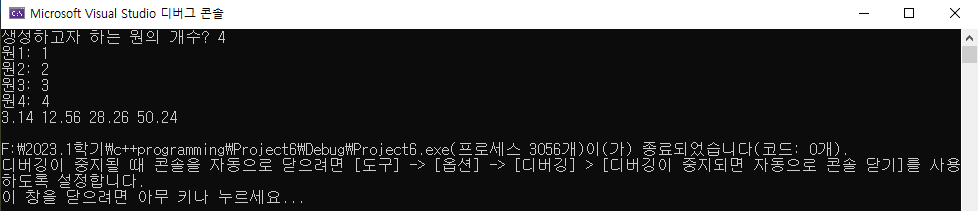
}

cout << endl;

delete[] p\_array;

}

🡪출력 결과



3. (예제 4-13) 문자열을 입역 받고 회전시키기.

#include <iostream>

#include <string>

using namespace std;

int main() {

string s;

cout << "아래에 문자열을 입력하세요. 빈 캉이 있어도 됩니다.(한글 안됨) " << endl;

getline(cin, s, '\n');

int len = s.length();

for (int i = 0; i < len; i++)

{

string first = s.substr(0, 1); //맨압 문자 1개

string sub = s.substr(1, len - 1);

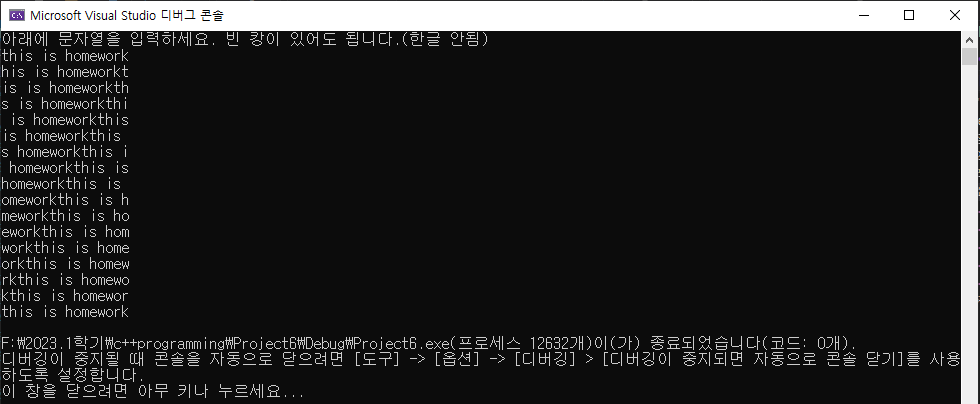
s = sub + first;

cout << s << endl;

}

}

🡪출력 결과



4. (예제 4-14) 문자열 처리 응용 – 덧셈 문자열을 입력 받아 덧셈 실행.

#include <iostream>

#include <string>

using namespace std;

int main() {

string s;

cout << "7+23+6+100+25와 같이 덧셈 문자열을 입력하세요. " << endl;

getline(cin, s, '\n');

int sum = 0;

int start\_index = 0;

while (true)

{

int f\_index = s.find('+', start\_index);

if (f\_index == -1) { // +를 더이상 발견 할수 없을 떄까지

string part = s.substr(start\_index);

if (part == "") {

break;

}

cout << part << endl;

sum += stoi(part); //문자열 수로 변환

break;

}

int count = f\_index - start\_index;

string part = s.substr(start\_index, count);

cout << part << endl;

sum += stoi(part);

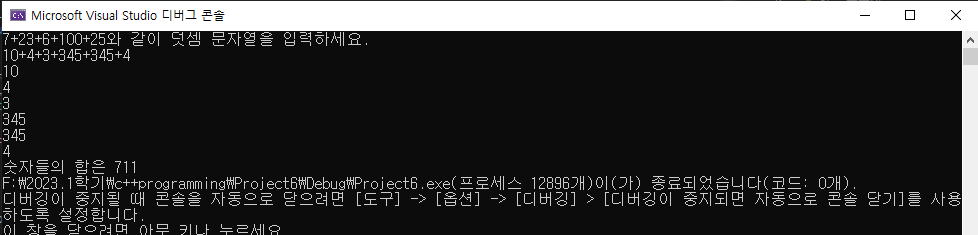
start\_index = f\_index + 1; //검색을 시작할 인덱스 전진

}

cout << "숫자들의 합은 " << sum;

}

🡪출력 결과



5. (예제 4-15) 문자열 find 및 replace.

#include <iostream>

#include <string>

using namespace std;

int main() {

string s;

cout << "여러 줄의 문자열을 입력하세요. 입력의 끝은 &문자입니다." << endl;

getline(cin, s, '&');

cin.ignore(); //'&' 뒤에 따라오는 엔터 제거 코드

string f, r;

cout << endl << "find: ";

getline(cin, f, '\n');

cout << "replace: ";

getline(cin, r, '\n');

int start\_index = 0;

while (true)

{

int f\_index = s.find(f, start\_index);

if (f\_index == -1)

break;

s.replace(f\_index, f.length(), r); //index부터 f의 길이만큼 r로 변경

start\_index = f\_index + r.length();

}

cout << s << endl;

}

🡪출력 결과

