**객체지향프로그래밍 LAB #08**

**<기초문제>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. 아래의 프로그램을 작성하시오. (/\*구현\*/ 부분을 채울 것, 표의 상단: 소스코드, 하단: 실행결과)

|  |
| --- |
| #include<iostream>  using namespace std;  //int &x = a; int &y = b; pass by reference  //int \*x = &a; int \*y = &b; pass by address  void swap(int\* x, int\* y) {  /\* 구현 \*/  }  int main() {  int a = 10, b = 20;  cout << a << ", " << b << endl;  swap(/\* 구현 \*/);  cout << a << ", " << b << endl;  return 0;  } |
|  |
| #include<iostream>  using namespace std;  //int &x = a; int &y = b; pass by reference  //int \*x = &a; int \*y = &b; pass by address  void swap(int\* x, int\* y) {  int temp = \*x;  \*x = \*y;  \*y = temp;  }  int main() {  int a = 10, b = 20;  cout << a << ", " << b << endl;  swap(&a, &b);  cout << a << ", " << b << endl;  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

2. 아래의 프로그램을 작성하시오. (/\*구현\*/ 부분을 채울 것)

|  |
| --- |
| #include<iostream>  using namespace std;  int sum(int x, int y) { return x + y; }  int mult(int x, int y) { return x \* y; }  // 함수 func의 포인터와 파라미터 x, y를 받아와서 함수 func의 실행값을 리턴해주는 함수  int evaluate(/\* 구현 \*/)  {  return func(x, y);  }  int main() {  int(\*func)(int, int);  /\* 구현 \*/ // sum 함수 주소값 대입  cout << func(10, 20) << endl;  /\* 구현 \*/ // mult 함수 주소값 대입  cout << func(10, 20) << endl;  cout << evaluate(&sum, 100, 200) << endl;  cout << evaluate(&mult, 100, 200) << endl;  return 0;  } |
|  |
| #include<iostream>  using namespace std;  int sum(int x, int y) { return x + y; }  int mult(int x, int y) { return x \* y; }  // 함수 func의 포인터와 파라미터 x, y를 받아와서 함수 func의 실행값을 리턴해주는 함수  int evaluate(int (\*func) (int, int), int x, int y) {  return func(x, y);  }  int main() {  int(\*func)(int, int);  func = sum;  cout << func(10, 20) << endl;  func = mult;  cout << func(10, 20) << endl;  cout << evaluate(&sum, 100, 200) << endl;  cout << evaluate(&mult, 100, 200) << endl;  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

3. 아래의 프로그램을 작성하시오. (/\*구현\*/ 부분을 채울 것)

|  |
| --- |
| #include<iostream>  #include<vector>  using namespace std;  void print(const vector<int>& v) {  for (unsigned i = 0; i < v.size(); i++)  cout << v[i] << "\t";  cout << endl;  }  int main() {  // 인덱스 0 1 2  vector<int> vec{ 10, 20, 30 };  print(vec);  /\* 구현 \*/ // vector 인덱스 0번에 직접 대입  /\* 구현 \*/ // vector의 at() 함수 이용  print(vec);  /\* 구현 \*/ // vector의 push\_back() 함수 이용  print(vec);  /\* 구현 \*/ // vector의 pop\_back() 함수 이용  print(vec);  cout << endl;  // 범위기반 for문을 이용한 vector 원소값 바꾸기 (슬라이드 p.26 참고)  for (/\* 구현 \*/) {  cout << "enter an element of vector v: ";  cin >> elem;  }  cout << endl << "[ vector v ]" << endl;  // 범위기반 for문을 이용한 화면 출력하기 (슬라이드 p.26 참고)  for (/\* 구현 \*/)  cout << elem << "\t";  cout << endl;  return 0;  } |
|  |
| #include<iostream>  #include<vector>  using namespace std;  void print(const vector<int>& v) {  for (unsigned i = 0; i < v.size(); i++)  cout << v[i] << "\t";  cout << endl;  }  int main() {  // 인덱스 0 1 2  vector<int> vec{ 10, 20, 30 };  print(vec);  vec[0] = 100;  vec.at(1) = 200;  print(vec);  vec.push\_back(400);  print(vec);  vec.pop\_back();  print(vec);  cout << endl;  // 범위기반 for문을 이용한 vector 원소값 바꾸기 (슬라이드 p.26 참고)  for (int& elem : vec) {  cout << "enter an element of vector v: ";  cin >> elem;  }  cout << endl << "[ vector v ]" << endl;  // 범위기반 for문을 이용한 화면 출력하기 (슬라이드 p.26 참고)  for (int elem : vec)  cout << elem << "\t";  cout << endl;  return 0;  } |

텍스트이(가) 표시된 사진

자동 생성된 설명

4. 아래의 프로그램을 작성하시오. (/\*구현\*/ 부분을 채울 것)

|  |
| --- |
| #include<iostream>  #include<iomanip>  #include<vector>  using namespace std;  bool is\_prime(int num) {  if (num < 2)  return false;  for (int i = 2; i < num; i++) {  /\* 구현 \*/ // num을 i로 나눈 나머지가 0이면 false 리턴  }  return true;  }  vector<int> primes(int lower, int upper) {  vector<int> v;  for (int i = lower; i <= upper; i++) {  /\* 구현 \*/ // 소수이면(is\_prime이 참이면) 뒤에 push  }  return v;  }  int main() {  int lower = 10, upper = 100;  /\* 구현 \*/ // 벡터 vec 선언 및 primes로 대입  for (int elem : vec)  cout << elem << setw(5);  cout << endl;  return 0;  } |
|  |
| #include<iostream>  #include<iomanip>  #include<vector>  using namespace std;  bool is\_prime(int num) {  if (num < 2)  return false;  for (int i = 2; i < num; i++) {  if (num % i == 0)  return false;  }  return true;  }  vector<int> primes(int lower, int upper) {  vector<int> v;  for (int i = lower; i <= upper; i++) {  if (is\_prime(i))  v.push\_back(i);  }  return v;  }  int main() {  int lower = 10, upper = 100;  vector<int> vec = primes(lower, upper);  for (int elem : vec)  cout << elem << setw(5);  cout << endl;  return 0;  } |

**텍스트, 화면이(가) 표시된 사진

자동 생성된 설명**

**<응용문제>\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. 정수 하나를 입력 받고, 해당 숫자의 제곱을 반환하는 함수를 만드시오.

* 함수의 반환형은 void형임.
* 함수의 인자로 &가 아닌 포인터를 사용. (pass by address)
* 입력은 main() 함수에서 받음.

1-출력화면:



|  |
| --- |
| #include <iostream>  using namespace std;  void square(int \*a) {  (\*a) = (\*a) \* (\*a);  }  int main() {  int n;  cout << "input> ";  cin >> n;  cout << "제곱값> ";  square(&n);  cout << n;  return 0;  } |

텍스트이(가) 표시된 사진

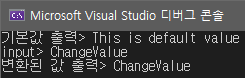
자동 생성된 설명

2. string 변수에 문자열을 입력 받아 저장하는 함수를 만드시오.

* main() 함수에서 string 변수 하나를 선언하고 기본값 설정.
* 변환할 문장은 main() 함수가 아닌 만든 함수에서 입력을 받음.
* 함수를 실행하기 전과 후의 string 변수를 출력.
* 함수의 반환형은 void형임.
* 함수의 인자로 &가 아닌 포인터를 사용. (pass by address)

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  /\* 함수 구현 \*/  int main() {  string str = "This is default value";  cout << "기본값 출력> " << str << endl;  /\* 함수 실행 \*/  cout << "변환된 값 출력> " << str << endl;  return 0;  } |

2-출력화면:



|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  void change(string \*a) {  cout << "input> ";  cin >> \*a;  }  int main() {  string str = "This is default value";  cout << "기본값 출력> " << str << endl;  change(&str);  cout << "변환된 값 출력> " << str << endl;  return 0;  } |

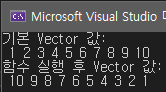
텍스트이(가) 표시된 사진

자동 생성된 설명

3. 크기가 10인 int형 vector를 선언하고, 각 인덱스 별로 1부터 10까지 할당한 후, vector 요소의 순서를 역순으로 바꾸는 함수를 만드시오.

* vector의 선언 및 초기화는 main() 함수에서 이뤄짐.
* 함수의 반환형은 void형임.
* 다른 vector 선언없이 처음 선언한 벡터의 요소를 뒤집어야 함. 즉, vector는 하나만 존재해야 함.

3-출력화면:



|  |
| --- |
| #include <iostream>  #include <vector>  using namespace std;  void change(vector<int>& vec\_a) {  int n = 10;  for (int i = 0; i < vec\_a.size(); i++) {  vec\_a[i] = n;  n--;  }  }    int main() {  vector<int> vec\_a{ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };  cout << "기본 Vector 값:" << endl;  for (int elem : vec\_a)  cout << elem << " ";  cout << endl;  cout << "함수 실행 후 Vector 값: " << endl;  change(vec\_a);    for (int elem : vec\_a)  cout << elem << " ";  return 0;  } |

텍스트이(가) 표시된 사진

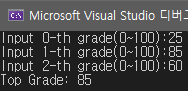
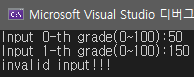
자동 생성된 설명

4. 성적 세 개를 입력하여 최고 점수를 출력하는 프로그램을 작성하시오. (주어진 코드를 이용하여 /\* 구현 \*/ 부분에 알맞은 코드를 작성하면 됨.)

* 사용자에게 0이상 100이하의 정수만 입력 받음. (이외의 입력을 받을 시 프로그램 종료)
* vector를 넘겨받아 최고 점수를 찾아 반환하는 topGrade함수의 반환형은 int형임.
* 참고) a>b이고 a>c이면 a가 최고 점수, a>b이고 a<c이면 c가 최고점수.

|  |
| --- |
| int inputGrade(float grade) {  cin >> grade;  if ((grade - (int)grade) != 0.0 || grade < 0 || grade > 100) {  cout << "invalid input!!!" << endl;  exit(EXIT\_FAILURE);  }  else  return (int)grade;  }  int topGrade(const vector<int>& grade) { /\* 구현 \*/ }  int main() {  vector<int> grades;  float \_grade = 0;  for(int i = 0; i < 3; i++) {  cout << "Input " << i << "-th grade(0~100):";  int grade = inputGrade(\_grade);  /\* 구현 \*/  }  /\* 구현 \*/  cout << "Top Grade: " << /\* 구현 \*/  return 0;  } |

4-출력화면:

|  |
| --- |
| #include <iostream>  #include <vector>  using namespace std;  int inputGrade(float grade) {  cin >> grade;  if ((grade - (int)grade) != 0.0 || grade < 0 || grade > 100) {  cout << "invalid input!!!" << endl;  exit(EXIT\_FAILURE);  }  else  return (int)grade;  }  int topGrade(const vector<int>& grade) {  int max;  if (grade[0] >= grade[1] && grade[0] >= grade[2])  max = grade[0];  else if (grade[1] >= grade[0] && grade[1] >= grade[2])  max = grade[1];  else if (grade[2] >= grade[0] && grade[2] >= grade[1])  max = grade[2];    return max;  }  int main() {  vector<int> grades;  float \_grade = 0;  for (int i = 0; i < 3; i++) {  cout << "Input " << i << "-th grade(0~100):";  int grade = inputGrade(\_grade);  grades.push\_back(grade);  }  int result = topGrade(grades);  cout << "Top Grade: " << result;  return 0;  } |

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**

**텍스트이(가) 표시된 사진

자동 생성된 설명**