|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cpp\_06\_객체포인터와동적생성 | 학번 : | 20175260 | 이름 : | 최영빈 |

* **개념 확인 – 객체 포인터와 배열, 동적 생성, 문자열**

1. 빈 괄호를 채워 넣으시오.
2. ( \* )연산자는 변수의 주소를 추출하기위해 사용한다
3. 동적으로 할당된 메모리를 반환하고자 할 때 (delete ) 연산자를 사용한다
4. new 연산자는 할당되는 동적 메모리의 ( 주소 )를 반환한다
5. 동적으로 할당되는 메모리는 ( 힙 )영역에서 할당 받는 메모리이다
6. delete 사용 시 ‘포인터 변수’는 반드시 ( new )를 이용하여 할당 받은 메모리의 주소여야 한다
7. 객체 포인터로 멤버를 접근할 때 ( -> ) 연산자를 사용한다
8. ( nullptr ) 은 null pointer를 의미하는 것으로 NULL 매크로 사용시 함수 매개변수로 전달하는 경우 int타입으로 추론되는 문제점을 해결할 수 있다
9. new 연산자를 사용하여 객체를 동적으로 할당할 때 ( 생성자 )를 호출한다.
10. 동적으로 할당 된 객체 소멸 시 ( 소멸자 )를 호출한다
11. 함수선언 시 ( const )를 사용하면 멤버 변수의 값을 변경할 수 없다
12. ( unique\_ptr ) 스마트 포인터는 공유할 수 없다
13. 스마트 포인터를 사용하려면 ( <memory> )헤더 파일이 필요하다
14. 다음 질문에 O, X로 답하시오
15. 동적 메모리 할당을 위해 new 함수를 사용한다 ( o )
16. 배열은 동적 할당 시 초기화를 할 수 없다( o )
17. 동적 메모리 반환 순서는 생성 순서와 동일해야 한다 ( x )
18. C++ 동적 메모리 할당 시 크기와 형 변환이 반드시 있어야 한다( x )
19. 객체 배열 생성 시 기본 생성자를 호출한다. ( o )
20. 동적으로 배열을 생성하면서 초기화 할 때 배열 크기는 생략할 수 있다 ( x )
21. 객체 포인터 변수는 초기화 없이 사용할 수 있다 ( x )
22. delete 연산자는 정적으로 할당된 메모리를 해제할 때도 사용할 수 있다 ( x )
23. 배열 형태로 동적 생성한 것은 배열 형태로 삭제해야 한다( o )
24. 클래스 멤버 변수에 대한 동적 생성은 생성자에서 할당하고 소멸자에서 동적 메모리를 해제해야 한다( o )
25. 스마트 포인터는 할당된 메모리를 자동으로 해제한다 ( o )
26. string\_view 클래스는 임시 객체를 생성하며 읽기 전용이다 ( x )
27. 스마트 포인터는 메모리누수와 같은 문제를 해결하기위해 사용한다. ( o )
28. 제시된 클래스에 대하여 질문에 답하시오

class Rec{

int w, h;

public:

int getW();

int getH();

Rec(){ }

Rec(int a, int b) : w(a), h(b){}

void write();

};

int main(){

Rec r(3,4);

}

1. Rec 클래스에 대한 포인터 변수 p를 선언하시오

Rec \*p

1. 선언된 포인터 변수 p에 객체 r의 주소를 지정하시오

p = &r;

1. 포인터 변수 p를 이용하여 write 함수를 호출할 수 있는 두 가지 방법을 제시하시오.

p->write();

(\*p).write();

1. 크기가 4인 Rec 객체 배열 arr를 동적으로 생성하는 문장을 new연산자와 공유할 수 없는 스마트 포인터를 사용하여 제시하시오

unique\_ptr<Rec[]> arr(Rec[4]);

Rec \*arr = new Rec[4];

1. 4번에서 new 연산자로 할당 받은 동적메모리를 반환하는 문장을 제시하시오

delete []arr;

1. 4번에서 생성된 배열에서 배열 원소 두번째에 저장된 객체의 write() 멤버를 참조하는 문장을 두 가지 방법으로 제시하시오. 단, 배열 원소 참조 시 []는 사용하지 말 것.

(\*(arr+1)).write();

(arr+1)->write();

1. 크기가 3인 Rec 객체 배열 dim을 동적으로 할당하면서 매개변수가 있는 생성자를 사용하여 초기화하는 문장을 제시하시오. 단, 초기화 값은 본인이 임의로 결정

Rec \*dim = new Rec[3]{Rec(1,2),Rec(2,3),Rec(3,4)};

1. 다음과 같이 초기화 된 객체 배열을 사용하여 멤버 함수 wirte를 호출하는 문장을 제시하시오. 단, 범위 기반 for를 사용할 것

Rec array[] = { Rec(13,6), Rec(5,8), Rec(3,12) };

for(auto data : array)

data.write();

1. 다음 프로그램의 실행 결과를 제시하시오.

|  |
| --- |
| #include <iostream> |
| using namespace std; |
| void fun (int\* x){ |
| cout << \*(x + 2); |
| } |
| int main (){ |
| int sample[] = {0, 10, 20, 30, 40}; |
| fun (sample); |
| return 0; |
| } |

20

|  |
| --- |
| int sample [5] = {5, 10, 15, 20, 25}; |
| cout << \*sample + 2 << endl; |
| cout << \*(sample + 2); |

1. 다음 코드의 실행 결과를 제시하시오.

7

15

1. 스마트 포인터 종류에 대하여 설명하시오

unique\_ptr, shared\_ptr, weak\_ptr

1. 다음과 같은 기능을 제공하는 string 클래스의 함수를 제시하시오

|  |  |
| --- | --- |
| 기능 | 함수 |
| 전체 문자열 삭제 | clear() |
| 일부 문자열 추출 | substr() |
| 문자열 연결 | append() |
| 부분 문자열 삭제 | erase() |
| 숫자를 문자열로 변환 | to\_string() |

* **개념 적용 응용 프로그래밍 – 객체 포인터와 배열 & 메모리 동적 할당**

1. 입력 받은 크기만큼 정수를 저장할 수 있는 공간을 할당 받고 10~99사이의 난수로 초기화 한 후 크기순으로 정렬하여 출력하는 프로그램을 작성하시오

|  |
| --- |
| [프로그램 소스]  #include <iostream>  #include <string>  #include <random>  using namespace std;  void sort(int a[], int b);  int main(){  random\_device rd;  mt19937 gen(rd());  uniform\_int\_distribution<int> dis(10,99);  int arraySize;  cout << "배열의 크기를 입력하세요 :";  cin >> arraySize;  int arr[arraySize];  for(int i = 0; i < arraySize; i++){  arr[i] = dis(gen);  }  sort(arr,arraySize);  for(auto data : arr){  cout << data << ", ";  }  return 0;  }  void sort(int array[],int size){  int temp;  for(int i = 0; i < size; i++){  for(int j = i+1; j <size; j++){  if(array[i] < array[j]){  temp = array[j];  array[j] = array[i];  array[i] = temp;  }  }  }  } |
| [실행 결과]  이미지 |

1. 다음과 같은 Sample 클래스가 있다. main()함수가 실행되도록 Sample클래스를 완성 하시오.

class Sample {

int \*p;

int size;

public:

Sample(int n); //n개 정수 배열의 동적 생성

void read(); // 동적 할당받은 정수 배열 p에 사용자로부터 정수를 입력 받음

void write(); // 정수 배열을 화면에 출력

int big(); // 정수 배열에서 가장 큰 수 리턴

~Sample(); // 소멸자

};

int main() {

Sample s(10); // 10개 정수 배열을 가진 Sample 객체 생성

s.read(); // 키보드에서 정수 배열 읽기

s.write(); // 정수 배열 출력

cout << "가장 큰 수는 " << s.big() << endl; // 가장 큰 수 출력

}

|  |
| --- |
| [프로그램 소스]    #include <iostream>  #include <string>  #include <memory>  using namespace std;  class Sample  {  int \*p;  int size;  public:  Sample(int n); //n개 정수 배열의 동적 생성  void read(); // 동적 할당받은 정수 배열 p에 사용자로부터 정수를 입력 받음  void write(); // 정수 배열을 화면에 출력  int big(); // 정수 배열에서 가장 큰 수 리턴  ~Sample(); // 소멸자  };  Sample::Sample(int n){  size = n;  p = new int[n];  }  void Sample::read(){  for(int i = 0; i < size; i++){  cout << i+1<<"번째 숫자를 입력하세요 : ";  cin >> p[i];  }  }  void Sample::write(){  for(int i = 0; i<size; i++){  cout <<p[i]<<", ";  }  cout <<"\n";  }  int Sample::big(){  int max = 0;  for(int i = 0; i<size; i++){  if(max < p[i])  max = p[i];  }  return max;  }  Sample::~Sample(){  delete []p;  }  int main()  {  Sample s(10); // 10개 정수 배열을 가진 Sample 객체 생성  s.read(); // 키보드에서 정수 배열 읽기  s.write(); // 정수 배열 출력  cout << "가장 큰 수는 " << s.big() << endl; // 가장 큰 수 출력  } |
| [실행 결과]  이미지 |

1. 다음과 같은 Custom 클래스가 있다. Custom 클래스와 main()함수를 작성하여 3개의 Custom객체를 갖는 배열을 선언하고 키보드에서 이름과 전화번호를 입력 받아 출력하고 검색하는 프로그램을 완성하시오

class Custom {

  string name;

  string tel;

public:

  Custom();

  string getName() { return name; }

  string getTel() { return tel; }

  void set(string name, string tel);

};

int main() {

  CustomManager manager(3);

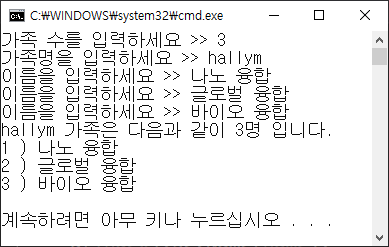
  manager.show();

  manager.search();

}

|  |
| --- |
| [프로그램 소스]  #include <iostream>  #include <string>  using namespace std;  class Custom{  string name;  string tel;  public:  Custom(){;}  string getName(){return name;}  string getTel() {return tel;}  void set(string name, string tel);  };  void Custom::set(string name,string tel){  this->name = name;  this->tel = tel;  }  class CustomManager{  Custom \*arr;  string name;  string tel;  int size;  public:  CustomManager(int n);  ~CustomManager();  void show();  void search();  };  CustomManager::CustomManager(int n){  arr = new Custom[n];  size = n;  for(int i=0; i<n; i++){  cout << i+1<<"번째 이름과 전화번호를 입력하세요";  cin >> name >>tel;  arr[i].set(name,tel);  }  }  CustomManager::~CustomManager(){  delete []arr;  }  void CustomManager::show(){  for(int i = 0; i<size; i++){  cout << i+1<<"번째 이름 :"<<arr[i].getName()<<" 전화번호 :"<<arr[i].getTel()<<'\n';  }  }  void CustomManager::search(){  bool a = false;  int b = 0;  cin >> name >>tel;  for(int i = 0; i < size; i++){  if(arr[i].getName() ==name && arr[i].getTel() ==tel){  a = true;  b = i;  }  }  if(a)  cout << b + 1 << "번째 이름과 전화번호와 동일합니다." << "\n";  else  cout << "일치하는 이름과 전화번호가 없습니다." << "\n";  }  int main(){  CustomManager manager(3);  manager.show();  manager.search();  } |
| [실행 결과]  이미지 |

1. Person, Family 클래스에 필요한 멤버를 추가하여 제시된 결과처럼 실행되도록 프로그램을 완성 하시오



class Person {

string name;

public:

Person(string name) { this->name = name; }

string getName() { return name; }

};

class Family {

Person\* p; // Person 배열 포인터

int size; // Person 배열의 크기. 가족 구성원 수

public:

Family(string name, int size); // size 개수만큼 Person 배열 동적 생성

void show(); // 모든 가족 구성원 출력

~Family();

};

int main() {

Family \*simpson;

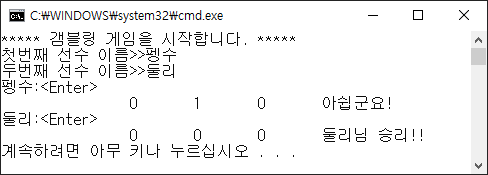
delete simpson;

}

|  |
| --- |
| [프로그램 소스]    #include <iostream>  #include <string>  using namespace std;  class Person{  string name;  public:  Person(){;}  Person(string name){this->name = name;}  string getName(){return name;}  void setName(string name){this->name = name;}  };  class Family{  Person\* p;  string fname;  string faname;  int size;  public:  Family();  Family(string name,int size);  void show();  ~Family();  };  Family::Family(){  cout << "가족 수를 입력하세요 >>";  cin >> size;  cout << "가족명을 입력하세요 >>";  cin >> fname;  Family(fname,size);  }  Family::Family(string name,int size){  fname = name;  this->size = size;  p = new Person[size];  for(int i = 0; i <size; i ++){  cout <<"이름을 입력하세요 >> ";  cin >> faname;  p[i].setName(faname);  }  show();  }  Family::~Family(){  delete []p;  }  void Family::show(){  cout << fname <<"가족은 다음과 같이"<<size<<"명 입니다."<<'\n';  for(int i = 0; i <size; i++){  cout << i+1<<" ) "<<p[i].getName()<<'\n';  }  }  int main()  {  Family \*simpson = new Family;  delete simpson;  return 0;  } |
| [실행 결과]  이미지 |

1. 갬블링 게임을 조건대로 작성하시오

* n명의 사람이 게임을 진행하며 선수 이름을 초기에 입력 받는다
* 선수가 번갈아 자신의 차례에서 <enter>키를 치면 랜덤한 3개의 수를 생성하고 3개의 수가 일치하면 승자가 된다
* 숫자의 범위는 1~3
* 선수는 Player 클래스로 작성하고 n명의 선수는 배열로 구성한다



|  |
| --- |
| [프로그램 소스]    #include <iostream>  #include <string>  #include <random>  #include <vector>  using namespace std;  random\_device rd;  mt19937 gen(rd());  uniform\_int\_distribution<int> dis(1, 3);  class Player{  string name;  string input;  bool flag = true;  int size;  int a,b,c;  vector<string> arr;  public:  Player(int n);  ~Player();  void playGame();  };  Player::Player(int n){  for(int i = 0; i < n;i++){  cout <<i+1<< "번째 선수 이름 >> ";  cin >> name;  arr.push\_back(name);  }  playGame();  }  Player::~Player(){  }  void Player::playGame(){  cin.ignore();  while(flag){  for (int i = 0; i < arr.size(); i++)  {  cout << arr[i] << ":<Enter>"<<'\n';  getline(cin,input);  a = dis(gen); b = dis(gen); c = dis(gen);  cout <<" "<<a<<" "<<b<<" "<<c<<" ";  if( a == b && b == c){  cout << arr[i]<<"님의 승리 !!";  flag = false;  break;  }  else  cout << "아쉽군요!"<<'\n';  }  }  }  int main(){  int value;  cout << "\*\*\*\*\* 갬블링 게임을 시작합니다. \*\*\*\*\*"<<"\n";  cout << "플레이 할 수를 입력하세요 : ";  cin >> value;  Player \*pl = new Player(value);  return 0;  } |
| [실행 결과]  이미지 |