|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **윈도우프로그래밍\_과제\_4주** | **이름 :** |  | **학번 :** |  |

* **내용 점검**

|  |
| --- |
| 1) 매개변수 배열 사용 예 |
| static void ParameterArray1(params object[] obj)  {  for (int i = 0; i < obj.Length; i++)  Console.WriteLine(obj[i]);  }  static void ParameterArray2(params int[] obj)  {  foreach( double temp in obj)  Console.Write(temp+"\t");  }  static void Main()  {  Random rnd = new Random(); //난수 생성을 위한 객체 생성  ParameterArray1(123, "Hello", true, 'A');  //rnd.Next()%10 : 0부터 9사이의 정수형 난수 생성  ParameterArray2(rnd.Next()%10, rnd.Next()%10, rnd.Next()%10);  } |
| **[실행결과]** |

|  |
| --- |
| 2) 명령행 인수 사용하기  프로젝트명-> 마우스 오른쪽 버튼 클릭 -> [속성] 선택    **이곳에 명령행 인수 작성** |
| static void Main(string[] args)  {  for (int i = 0; i < args.Length; ++i)  Console.WriteLine("Argument[{0}] = {1}", i, args[i]);  } |
| **[실행결과]** |

|  |
| --- |
| 3) 소멸자& 정적 생성자& 생성자 사용 예 |
| class Sample {  string cname;  public static int value;  public Sample(string cname)  {  this.cname = cname;  }  static Sample() //정적 생성자  {  value = 10;  Console.WriteLine("정적 생성자 호출");  }  ~Sample()  {  Console.WriteLine(cname + "객체를 소멸합니다");  }  }  class Program {  static void Main(string[] args)  {  Console.WriteLine("첫 번째 위치");  Console.WriteLine(Sample.value);  Console.WriteLine("두 번째 위치");  Sample sample1 = new Sample("destructor1");  Sample sample2 = new Sample("destructor2");  Console.WriteLine("세 번째 위치");  }  } |
| **[실행결과]** |

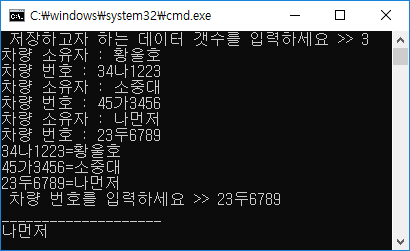
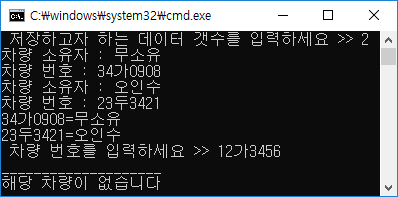
|  |
| --- |
| 4) 프로퍼티 사용 예 |
| class Fraction{  private int numerator;  private int denominator;  public int Numerator  {  get { return numerator; }  set { numerator = value; } //value는 예약된 키워드  }  public int Denominator  {  get { return denominator; }  set { denominator = value; }  }  override public string ToString() //객체 내용을 문자열로 반환  {  return numerator + "/" + denominator;  }  public string Message  {  get { return Console.ReadLine(); }  set { Console.WriteLine(value); }  }  }  class Program{  static void Main(string[] args)  {  Fraction f1 = new Fraction();  int i;  f1.Numerator = 1; // invoke set-accessor in Numerator  i = f1.Numerator + 1; // invoke get-accessor in Numerator  f1.Denominator = i; // invoke set-accessor in Denominator  Console.WriteLine(f1.ToString());  Fraction f2 = new Fraction();  f2.Message = f2.Message;  }  } |
| **[실행결과]** |

|  |
| --- |
| 5) 인덱서 사용 예1 |
| class DayCollection{ // Using a string as an indexer value  string[] days = { "일", "월", "화", "수", "목", "금", "토" };  // This method finds the day or returns -1  private int GetDay(string testDay){  for (int j = 0; j < days.Length; j++) {  if (days[j] == testDay) {  return j; }  }  return 999;  }  public int this[string day] {// The get accessor returns an integer for a given string  get { return (GetDay(day)); }  }  }  class Program{  static void Main(string[] args) {  DayCollection week = new DayCollection();  System.Console.WriteLine(week["일"]);  System.Console.WriteLine(week["Made-up Day"]);  }  } |
| **[실행결과]** |

|  |
| --- |
| 7) 인덱서 사용 예2 |
| class MyDictionary {  string[] storage = new string[3];  public MyDictionary(){  for (int i = 0; i < storage.Length; i++){  storage[i] = string.Format("item{0}=value{0}", i, i);  }  }  public string this[int index]{ //정수를 매개변수로 받는 인덱서  get{  if (AvailIndex(index)){ //유효한 인덱스일 때  return GetValue(storage[index]); //storage[index] 요소의 값을 반환  }  return string.Empty;  }  }  public string this[string key]{ //문자열을 매개변수로 받는 인덱서  get {  string element = FindKey(key); //키에 해당하는 요소 문자열을 찾는다.  return GetValue(element); //요소 문자열에서 값을 추출하여 반환한다.  }  }  private bool AvailIndex(int index){  return (index >= 0) && (index < storage.Length);  }  private string FindKey(string key) {  foreach (string s in storage){ //보관된 각 요소에 대해 반복 수행  if (s.IndexOf(key) == 0){ //요소 문자열이 key로 시작할 때  return s; //요소 문자열 반환  }  }  return string.Empty;  }  private string GetValue(string s) {  int index = s.IndexOf('='); //'='문자가 시작되는 위치를 찾는다.  return s.Substring(index + 1); //index+1 뒤에 있는 부분 문자열을 반환한다.  }  public int Size{ //요소를 보관하는 storage의 크기를 반환  get{  return storage.Length;  }  }  }  class Program {  static void Main(string[] args)  MyDictionary md = new MyDictionary();  for (int i = 0; i < md.Size; i++){  Console.WriteLine(md[i]); //정수를 인자로 받는 인덱서 사용  }  Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");  Console.WriteLine(md["item1"]); //문자열 인자로 받는 인덱서 사용  }  } |
|  |

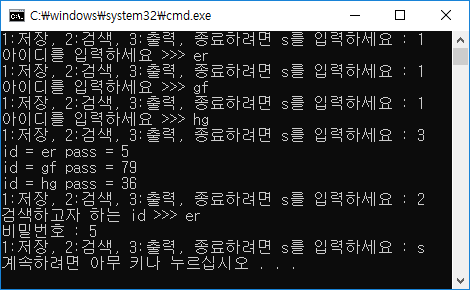
* 프로그램 과제

1. 차량 소유자를 검색하여 결과를 출력하는 프로그램을 제시된 결과처럼 실행 될 수 있도록 완성하시오. (힌트: 내용점검 7번)

|  |
| --- |
| static void Main(string[] args) {  Car md;  Console.Write(" 저장하고자 하는 데이터 갯수를 입력하세요 >> ");  md = new Car(int.Parse(Console.ReadLine()));  for (int i = 0; i < md.Size; i++) {  Console.WriteLine(md[i]); //정수를 인자로 받는 인덱서 사용  }  Console.Write(" 차량 번호를 입력하세요 >> ");  string result = md[Console.ReadLine()];  Console.WriteLine("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");  if(result != string.Empty)  Console.WriteLine(result); //문자열 인자로 받는 인덱서 사용}  else  Console.WriteLine("해당 차량이 없습니다");  }  **프로그램 소스** |
|  |
| **[실행결과]** |

1. 제시된 결과처럼 아이디와 비밀번호를 관리하는 프로그램을 완성하시오. S를 입력하면 프로그램이 종료되며 대소문자를 구분하지 않는다



class Person **//Person 클래스에 대한 생성자 없음**

{

int pass; //비밀번호

string id; //아이디

//비밀번호와 아이디에 대한 프로퍼티 정의 – 본인작성

}

class PersonManage

{

Person[] people;

int index = 0;

Random rnd;

//크기가 50인 Person 배열 객체 생성하고 난수를 위한 객체를 생성하는 생성자 – 본인 작성

//아이디를 입력받으며 비밀번호는 1부터 100사이의 난수를 생성하여 저장 하는 메소드 – 본인작성

//문자열 id를 매개변수로 받아 id에 해당하는 비밀번호를 반환하는 인덱서 – 본인작성

//매개변수로 받은 문자열 id와 일치하는 비밀번호 반환, 없으면 0을 반환하는 메소드 – 본인작성

//배열에 저장된 모든 내용을 출력하는 메소드 – 본인작성

}

class Program

{

static void Main(string[] args)

{

// PersonManage 객체를 생성하고 제시된 결과처럼 입력된 메뉴를 처리 – 본인작성

}

}

|  |
| --- |
| **프로그램 소스** |
|  |
| **[실행결과]** |

1. 교재 217p 문제 4.11

|  |
| --- |
| **프로그램 소스** |
|  |
| **[실행결과]** |

1. 교재 217p 문제 4.13

|  |
| --- |
| **프로그램 소스** |
|  |
| **[실행결과]** |