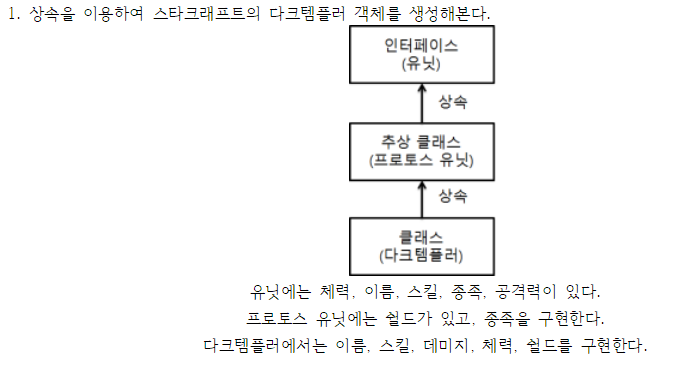
**윈도우즈 프로그래밍**

**컴퓨터공학과**

**20204062 이인규**

****

**소스 코드 - class ProtossUnit**

|  |
| --- |
| class ProtossUnit{  public int strength = 40;  public String skill = "클로킹";  public int damage = 45;  public int shield = 80;  public String clan;  public String name;  public ProtossUnit(String clan, String name){  this.clan = clan;  this.name = name;  }  public int getDamage(){  return damage;  }  public int getStrength(){  return strength;  }  public int getShield(){  return shield;  }  public String getClan(){  return clan;  }  public String getName(){  return name;  }  public String getSkill(){  return skill;  }  } |

**소스 코드 설명**

strength, skill, damage, shield 값들은 기본적으로 설정되어있는 값들이므로 선언과 동시에 초기화 한다.

생성자를 통해서 clan, name의 값을 받아오며, 각 멤버 변수들의 Getter를 생성한다.

**소스 코드 class DarkTemplar**

|  |
| --- |
| class DarkTemplar : ProtossUnit{  public DarkTemplar(String clan, String name) : base(clan, name){  }  } |

**소스 코드 설명**

name 값만 가지고 있으며, 부모 클래스인 ProtossUnit의 멤버 변수들과 생성자를 그대로 사용한다.

**소스 코드 class Nexus**

|  |
| --- |
| class Nexus{  int mineral;  int gas;  public Nexus(int mineral){  this.mineral = mineral;  if(this.mineral >= 50){  Console.WriteLine("프로브 생산");  }  else{  Console.WriteLine("광물이 부족합니다.");  }  }  public Nexus(int mineral, int gas){  this.mineral = mineral;  this.gas = gas;  if(this.mineral >= 300 && this.gas >= 300){  Console.WriteLine("모선 생산");  }  else{  Console.WriteLine("광물 혹은 가스가 부족합니다.");  }  }  } |

**소스 코드 설명**

광물과 가스를 생성자로부터 받아온다. 그 후 프로브를 생산할 수 있는지와 모선을 생산할 수 있는지 확인 후 정해진 출력문을 출력한다.

**소스 코드 class program**

|  |
| --- |
| class program{  static void Main(String[] args){  ProtossUnit DT = new DarkTemplar("프로토스", "다크템플러");  int Damage = DT.getDamage();  int Strength = DT.getStrength();  int Shield = DT.getShield();  Console.WriteLine("종족: " + DT.getClan());  Console.WriteLine("이름: " + DT.getName());  Console.WriteLine("쉴드: " + Shield);  Console.WriteLine("체력: " + Strength);  Console.WriteLine("데미지: " + Damage.ToString());  Console.WriteLine("스킬: " + DT.getSkill());  Console.WriteLine();  Nexus N1 = new Nexus(40);  Nexus M1 = new Nexus(30, 30);  Console.WriteLine();  Nexus N2 = new Nexus(50);  Nexus M2 = new Nexus(300, 300);  Console.WriteLine();  }  } |

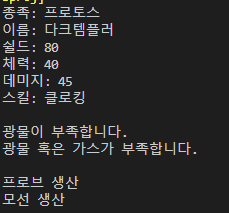
**소스 코드 설명**

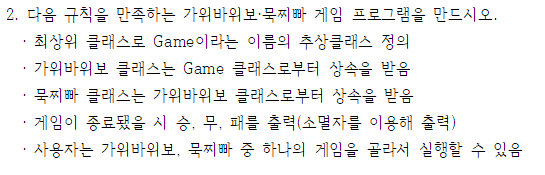
설명에 나와 있는 대로 출력할 수 있도록 DT 인스턴스를 생성한다. 이때 ProtosUnit 데이터형을 가지고 있는 DT 변수에는 DarkTemplar의 객체를 생성한다.

**전체 소스 코드**

|  |
| --- |
| class ProtossUnit{  public int strength = 40;  public String skill = "클로킹";  public int damage = 45;  public int shield = 80;  public String clan;  public String name;  public ProtossUnit(String clan, String name){  this.clan = clan;  this.name = name;  }  public int getDamage(){  return damage;  }  public int getStrength(){  return strength;  }  public int getShield(){  return shield;  }  public String getClan(){  return clan;  }  public String getName(){  return name;  }  public String getSkill(){  return skill;  }  }  class DarkTemplar : ProtossUnit{  public DarkTemplar(String clan, String name) : base(clan, name){  }  }  class Nexus{  int mineral;  int gas;  public Nexus(int mineral){  this.mineral = mineral;  if(this.mineral >= 50){  Console.WriteLine("프로브 생산");  }  else{  Console.WriteLine("광물이 부족합니다.");  }  }  public Nexus(int mineral, int gas){  this.mineral = mineral;  this.gas = gas;  if(this.mineral >= 300 && this.gas >= 300){  Console.WriteLine("모선 생산");  }  else{  Console.WriteLine("광물 혹은 가스가 부족합니다.");  }  }  }  class program{  static void Main(String[] args){  ProtossUnit DT = new DarkTemplar("프로토스", "다크템플러");  int Damage = DT.getDamage();  int Strength = DT.getStrength();  int Shield = DT.getShield();  Console.WriteLine("종족: " + DT.getClan());  Console.WriteLine("이름: " + DT.getName());  Console.WriteLine("쉴드: " + Shield);  Console.WriteLine("체력: " + Strength);  Console.WriteLine("데미지: " + Damage.ToString());  Console.WriteLine("스킬: " + DT.getSkill());  Console.WriteLine();  Nexus N1 = new Nexus(40);  Nexus M1 = new Nexus(30, 30);  Console.WriteLine();  Nexus N2 = new Nexus(50);  Nexus M2 = new Nexus(300, 300);  Console.WriteLine();  }  } |

**소스 코드 실행 결과**

****

****

**소스 코드 class Game**

|  |
| --- |
| abstract class Game{  protected static int userWin=0, computerWin=0, drawMatch=0;  public Game(){  }  ~Game(){  recordPrint();  }  public abstract void result();  public void recordPrint(){  Console.WriteLine(userWin + "승 " + computerWin + "패 " + drawMatch + "무승부입니다.");  }  } |

**소스 코드 설명**

추상 클래스로 선언된 Game은 소멸자를 가지고 있으며, 소멸할 경우 recordPrint()를 출력하게 된다.

**소스 코드 class Srp**

|  |
| --- |
| class Srp : Game{  public int shot;  public int user;  public Srp(){  Random rand = new Random();  this.shot = rand.Next(1,4);  }  public void play(){  Console.Write("입력하세요 [가위(1), 바위(2), 보(3), 종료(0)] : ");  int i = Convert.ToInt32(Console.ReadLine());  if(i == 0){  return;  }  this.user = i;  result();  }  public int check(){  if(user != 1 && shot > user){  return -1; // 졌음  }  else if(shot == user){  return 0; // 비겼음  }  else if(user == 1 && shot == 3){  return 1; // 이겼음  }  else{  return 1; // 이겼음  }  }  public override void result(){  int chk = check();  String print;  if(shot == 1){  print = "컴퓨터는 가위를 냈습니다.";  }  else if(shot == 2){  print = "컴퓨터는 바위를 냈습니다.";  }  else{  print = "컴퓨터는 보를 냈습니다.";  }  Console.WriteLine(print);  if(chk == 0){  print = "비겼습니다.";  drawMatch += 1;  }  else if(chk == -1){  print = "졌습니다.";  computerWin += 1;  }  else{  print = "이겼습니다.";  userWin += 1;  }  Console.WriteLine(print);  Console.WriteLine();  }  } |

**소스 코드 설명**

주요 메소드를 설명해보면, check() 메소드로 유저가 이겼는지 판단해준다. 그후 result() 메소드를 이용해 승패 여부를 출력해준다.

**소스 코드 class Mjb**

|  |
| --- |
| class Mjb : Srp{  int attacter = -1;  Random rand = new Random();  public Mjb(){  shot = rand.Next(1,4);  }  public void play(){  Console.Write("입력하세요 [가위(1), 바위(2), 보(3), 종료(0)] : ");  int i = Convert.ToInt32(Console.ReadLine());  shot = rand.Next(1,4);  if(i == 0){  return;  }  user = i;  result();  }  public override void result(){  int chk = check();  String print;  if(shot == 1){  print = "컴퓨터는 가위를 냈습니다.";  }  else if(shot == 2){  print = "컴퓨터는 바위를 냈습니다.";  }  else{  print = "컴퓨터는 보를 냈습니다.";  }  Console.WriteLine("check: " + chk);  Console.WriteLine(print);  if(attacter == -1){  if(chk == 0){  Console.WriteLine("비겼습니다.");  Console.WriteLine();  play();  }  else if(chk == -1){  Console.WriteLine("공격자는 컴퓨터입니다.");  Console.WriteLine();  attacter = 0;  play();  }  else{  Console.WriteLine("공격자는 유저입니다.");  Console.WriteLine();  attacter = 1;  play();  }  }  else if(attacter == 1){  if(chk == 0){  Console.WriteLine("이겼습니다.");  Console.WriteLine();  userWin += 1;  }  else if(chk == -1){  Console.WriteLine("공격자는 컴퓨터입니다.");  Console.WriteLine();  attacter = 0;  play();  }  else{  Console.WriteLine("공격자는 유저입니다.");  Console.WriteLine();  play();  }  }  else{  if(chk == 0){  Console.WriteLine("졌습니다.");  Console.WriteLine();  computerWin += 1;  }  else if(chk == -1){  Console.WriteLine("공격자는 유저입니다.");  Console.WriteLine();  attacter = 1;  play();  }  else{  Console.WriteLine("공격자는 컴퓨터입니다.");  Console.WriteLine();  play();  }  }  }  } |

**소스 코드 설명**

묵찌빠 클래스이며, 매 반복마다 묵/찌/빠를 낼지 결정한다. 이전에 상속 받은 Srp 클래스의 check() 메소드를 이용해 승/무/패 여부를 판단하고, 현재 공격자가 누구인지에 따라서 다른 출력문을 출력해준다.

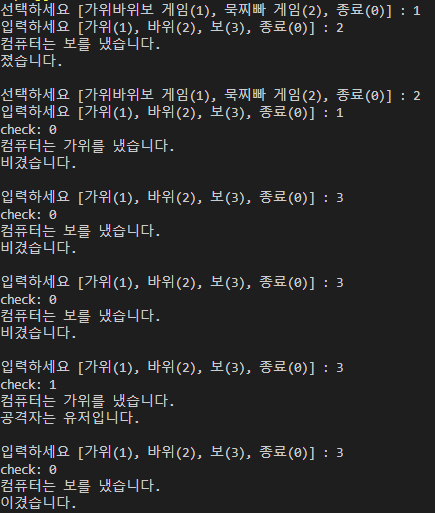
**소스 코드 class Program**

|  |
| --- |
| class Program{  static void Main(String[] args){  while(true){  Console.Write("선택하세요 [가위바위보 게임(1), 묵찌빠 게임(2), 종료(0)] : ");  int i = Convert.ToInt32(Console.ReadLine());  if(i == 1){  new Srp().play();  }  else if(i == 2){  new Mjb().play();  }  else if(i == 0){  break;  }  else{  Console.WriteLine("잘못 입력하셨습니다.");  }  }  }  } |

**소스 코드 설명**

0을 입력받으면, break; 문을 통해서 while문을 빠져나오게 되지만, 다른 입력을 받으면, 해당 프로그램을 실행합니다. 이때 1은 가위바위보의 play() 메소드를 실행시키며, 2은 묵찌빠의 메소드를 실행시킵니다.

**실행 결과**

****