1. 정수형식

* 프로그렘을 구성하는 요소 : 데이터 + 로직
* byte(1바이트 0~255), short(2바이트 -3만~3만), int(4바이트 -21억~21억), long(8바이트)
* sbyte(1바이트 -128~127), ushort(2바이트 0~6만), uint(4바이트 0~43억), ulong(8바이트)
* 변수 형식이 커질경우 메모리 사용이 커지기 때문에 알맞은 변수 버전을 사용해야 함

1. 진법(2, 10, 16 진수)

* 10진법(0~9 00), 2진법(0~1 0b00), 16진법(0~F 0x0F)
* 2진법 => 16진법은 4묶음으로 바꾸면 편함 (0b 1111 1001 => 0xF9)

1. Boolean, 소수, 문자, 문자열

* bool : 참/거짓을 나타내는 변수(true, false) – 1바이트
* float(4바이트), double(8바이트) : 소수를 나타내는 변수
* char(2바이트) : 딱 하나의 문자를 나타내는 변수(ex) ‘a’)
* string : char들의 집합체(“Hellow World”)

1. 형식 변환

* 주의해야 할 점

1. 변수의 크기가 다른 경우(int => short로 변환)

* int a = 100; short b = (short)a; //변수의 크기를 맞추기 위치 (변수)를 사용해 “캐스팅”
* 주의 : 크기가 큰 변수 => 작은 변수로 캐스팅시 데이터 소실이 일어날 수 있음

1. 변수의 크기는 같은데, 부호가 다를 경우

* Byte c = 255; sbyte sb = (sbyte)c;
* 이때 sb의 값은 255가 아닌 -1이 된다. Sbyte의 첫 부호는 음수를 나타내기 때문(오버, 언더 플로우)

1. 데이터 연산

* 산술 연산자 : +, -, \*, /, %
* 증가(감소) 연산자 : ++, -- //++hp, hp++ 차이 => 실행 시점
* 비교 연산자 : <, >, ==(같다), !=(다르다)
* 논리 연산자 : &&(and), ||(or), !(not)

1. 비트연산

* 쉬프트 연산 : <<, >> 2진수(bit)에서 자리를 왼쪽 또는 오른쪽으로 옮겨줌
* 3<<5 는 0000 0011 에서 5자리 밀려난, 0110 0000 (96)
* &(and), |(or), ^(xor), ~(not)
* 그럼 이 비트 연산을 어디에서 사용하는가? : 게임에서 아이템의 id를 구현할 때는 n비트씩 쪼개서 정보를 넣는 경우가 많다(ex) 앞의 4비트는 아이템 유형 바로 뒤의 4비트는 드랍 지역 …) 이럴때, 0000(4비트)에 해당하는 정보를 변수에 넣고, (32비트라면) 0000 << 28로 정보를 넘길 수 있다.
* Xor는 암호화에서 자주 사용된다. (숫자에 똑같은 값을 xor 하면 원래 값과 같은 값이 나온다)

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 할당연산

* 간단하게 나타내기(ex) a = a + 1 은 a += 1과 같은 의미이다. a >>= 1, a &= 1 등 다른 연산도 동일)
* Var : 컴파일러가 알아서 변수를 찾아줌 (편리하지만 남용하면 안된느 함수)

1. If문

* If (조건문) {참;} else {거짓;}

1. Switch 문

* Switch(조건문) case (조건): break; default:
* Break는 스위치 문을 빠져나옴, default는 if의 else와 같은 의미

1. 삼항연산자

* ?를 사용해서 if문처럼 사용 가능
* bool isPair = ((num%2)==0 ? true : false);

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 가위바위보 코딩