**Report**

**For java Programming**

**학과 :**

**학번 :**

**이름 :**

컴퓨터공학과

20184071

김도현

**Println 메소드를 사용하여 별 찍기**

1. 프로그램 소스코드 및 설명과 결과

**public** **class** PrintStars { //자바는 클래스에서 작동되므로 클래스 선언

**public** **static** **void** main(String arg[]) { //c 언어에서 main 함수와 같은 역할을 하는 main 메소드 생성

System.out.println(" \*"); //println 하면 자동으로 개행된다

System.out.print(" \*\*\*\n"); // print 를 하면 자동으로 개행되지 않으므로 개행문자를 넣어준다.

System.out.println(" \*\*\*"

+ "\*\*"); // + 연산자를 사용하여 String 이 길 경우 짤라서 사용도 가능하다

System.out.print(" \*\*\*"

+ "\*\*\*\*\n"); //print 메소드도 마찬가지로 + 연산자를 사용하여 나눠쓸수 있다.

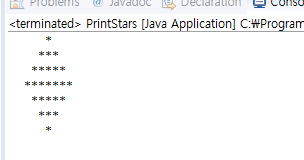
System.out.println(" \*\*\*\*\*");

System.out.println(" \*\*\*");

System.out.println(" \*");

}

}



실행결과 )) 별이 정상적으로 잘 찍힌다.

**다양한 형태의 변수 출력**

1.프로그램 소스코드 및 설명과 결과

**public** **class** PrintVariables {

**public** **static** **void** main(String arg[]) {//c 언어에서 main 함수와 같은 역할을 하는 main 메소드 생성

**int** myNum = 10; //int 형 변수 myNum 샌성

**float** myFloatNum = 3.145f; //float 형 변수 myFloatNum 생성, 변수 뒤에 f 를 붙여야한다

**char** myLetter = 'A'; // char 형 변수 myletter 생성

**boolean** myBool = **true**; //참 거짓을 판별하는 myBool 변수 생성

String myString = "HelloWorld"; //문자열을 뜻하는 변수 String 형 myString 생성

System.out.println("myNum = " + myNum); //int 형 변수 출력

System.out.println("myFloatNum = " + myFloatNum); //float 형 변수 출력

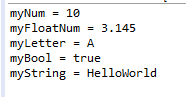
System.out.println("myLetter = " + myLetter); //char 형 변수 출력

System.out.println("myBool = " + myBool); //boolean 형 변수 출력

System.out.println("myString = " + myString); //String 형 변수 출력

}

}



실행결과 )) println 메소드에 형식지정자 지정할 필요 없이 출력이 가능하다.

**For 문을 사용한 소수 판별하기**

1. 프로그램 소스코드 및 설명과 결과

**public** **class** CheckPrimeUseFor {

**public** **static** **void** main(String args[]) { //c 언어에서 main 함수와 같은 역할을 하는 main 메소드 생성

**int** num = 29; //소수인지 아닌지 판별할 수

Boolean isPrime = **true**; //is Prime

**for** (**int** i = 2 ; i < num ; i ++) { //2~자기 자신 - 1 까지 했을때 나누어 떨어지는지 안떨어지는지 판별해야 함

**if** (num%i == 0) { //num 을 i 로 나눴을때 나눠 떨어지면 그건 소수가 아니다

isPrime = **false**; //그러므로 isPrime 을 거짓으로 바꾼다

**break**; //더이상 할 가치가 없으므로 break 를 걸어 for 루프 탈출

}

}

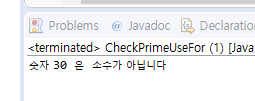
System.out.println("숫자 " + num +" 은 "+ (isPrime ? "소수 입니다": "소수가 아닙니다") );

//3항 연산자를 사용하여 isPrime 이 참이면 "소수입니다" 출력 거짓이면 "소수가 아닙니다" 출력

}

}





실행결과 )) 소수는 소수라 출력되고 소수가 아니면 소수가 아니라고 출력된다.

**While 문을 사용한 소수 판별**

1. 프로그램 소스코드 및 설명과 결과

**public** **class** CheckPrimeUseWhile {

**public** **static** **void** main(String args[]) { //c 언어에서 main 함수와 같은 역할을 하는 main 메소드 생성

**int** num = 29; //소수인지 아닌지 판별할 수

Boolean isPrime = **true**; //is Prime

**if** (num == 1) { //1은 소수이므로 main 메소드 종료

System.out.println("숫자 " + num +" 은 소수 입니다");

**return**;

}

**int** i = 2; //while 루프를 돌리기 위한 변수 선언

**while** (i < num) { //2~자기 자신 - 1 까지 했을때 나누어 떨어지는지 안떨어지는지 판별해야 함

**if** (num%i == 0) { //num 을 i 로 나눴을때 나눠 떨어지면 그건 소수가 아니다

isPrime = **false**; //그러므로 isPrime 을 거짓으로 바꾼다

**break**; //더이상 할 가치가 없으므로 break 를 걸어 while 루프 탈출

}

i++; //while 문이 끝나기 전 i 에 1을 더해준다

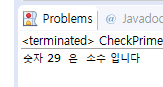
}

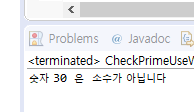
System.out.println("숫자 " + num +" 은 "+ (isPrime ? "소수 입니다": "소수가 아닙니다") );

//3항 연산자를 사용하여 isPrime 이 참이면 "소수입니다" 출력 거짓이면 "소수가 아닙니다" 출력

}

}





실행결과 )) for 문대신 while 문을 사용했으므로 소수는 소수. 소수가 아니면 소수가 아니라고 출력된다.

**숫자를 두개의 소수의 합으로 출력**

1. 프로그램 소스코드 및 설명과 결과

**public** **class** PlusPrimeTwoNum {

**public** **static** **void** main(String[] args) {

**int** num = 34 ; //두 소수로 할지 말지 값 저장

**for** (**int** i = 2; i < num; i ++) { //num - 1 까지 반복하여 그 범위 안에 있는 소수들을 구한다.

**if** (*CheckPrime*(i)) { //checkPrime 함수를 사용하여 소수인지 구한다

//반환값이 참 거짓 밖에 없다

//i 가 소수면 아래 코드가 실행된다

**for** (**int** j = 2; j < num ; j ++) { //j 가 2 ~ num -1 까지 반복하여 소수 구간을 한번 더 찾는다

**if** (*CheckPrime*(j) && j + i == num) { //j 가 소수고 i 와 j 가 더한 값이 num 이면

System.out.println(num + " = " + i + " + " + j); //두개 더한 값을 출력

//전 단계로 인해 i 는 무조건 소수

}

}//for (int j = 2; j < num ; j ++)

}//if (CheckPrime(i))

}

}

**public** **static** **boolean** CheckPrime(**int** num) { //소수인지 아닌지 판별할 메소드 생성, 2번 이상 사용하기 때문에 함수로 만든다.

Boolean isPrime = **true**; //is Prime

**for** (**int** i = 2 ; i < num ; i ++) { //2~자기 자신 - 1 까지 했을때 나누어 떨어지는지 안떨어지는지 판별해야 함

**if** (num%i == 0) { //num 을 i 로 나눴을때 나눠 떨어지면 그건 소수가 아니다

isPrime = **false**; //그러므로 isPrime 을 거짓으로 바꾼다

**break**; //더이상 할 가치가 없으므로 break 를 걸어 for 루프 탈출

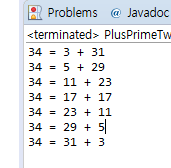
}

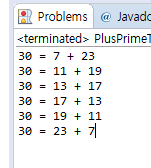
}

**return** isPrime; //반환값이 boolean 이므로 참 거짓 반환을 해준다.

}

}





실행결과 )) 숫자를 바꿔도 정상적으로 출력된다. 알고리즘을 입력수 – 소수 가 소수인지 아닌지 판별하는식으로 바꾸면 성능이 개선될것으로 보인다.

**원주율 구하기**

1. 프로그램 소스코드 및 설명과 결과

**public** **class** GetPI {

**public** **static** **void** main(String[] args) {

**double** pi = 3.0; //처음은 3 으로 시작해야 한다

**double** plus; //소수점 자리를 구할 변수 선언

**for** (**int** i = 1; i < 1000; i ++) { //1000회 반복한다

// 분모는 2n \* 2n + 1 \* 2n + 2 이다

plus = ( (2\*i) )\*( (2\*i) +1 )\*( (2\*i)+2 );

//더하는 값은 4 를 그 전에 구한 2n \* 2n +1 \* 2n+2 로 나눠야 한다.

plus = 4 / plus;

//i 가 짝수면 구한 값을 빼야 하므로 plus 에 - 를 붙인다

**if** ( i % 2 == 0) {

plus = -plus;

}

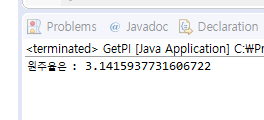
pi += plus; //모든 연산이 끝나면 pi 에 plus 를 더해준다

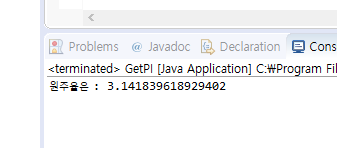
}

System.out.println("원주율은 : " + pi); //구한 값을 출력한다.

}

}

 (1000 번 for 문 했을시)

(100 번 했을 시)

실행결과 )) for 문의 횟수에 따라 원주율의 정확도가 달라짐을 알 수 있다.