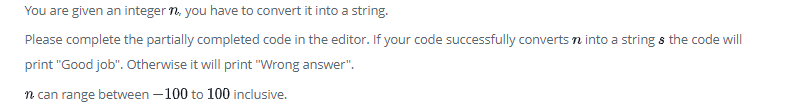
**Easy**

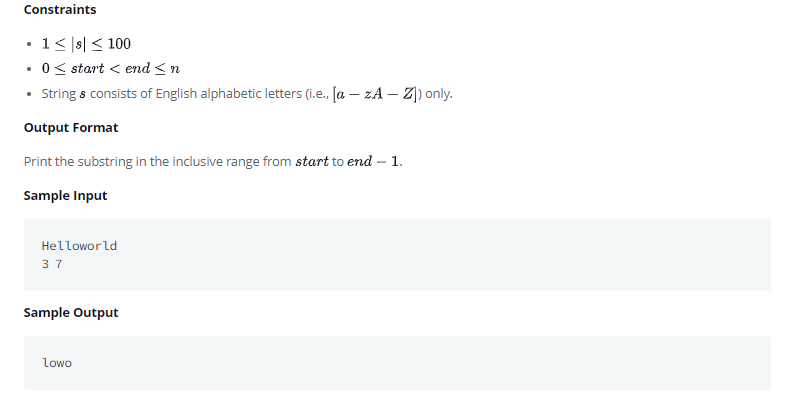
1. Java Int to String

<https://www.hackerrank.com/challenges/java-int-to-string/problem>



1. **import** java.util.\*;
2. **import** java.security.\*;
3. **public** **class** Solution {
4. **public** **static** **void** main(String[] args) {
5. DoNotTerminate.forbidExit();
6. **try** {
7. Scanner in = **new** Scanner(System.in);
8. **int** n = in .nextInt();
9. in.close();
10. *//String s=???; Complete this line below*
11. *//Write your code here*
12. String s = String.valueOf(n);
14. **if** (n == Integer.parseInt(s)) {
15. System.out.println("Good job");
16. } **else** {
17. System.out.println("Wrong answer.");
18. }
19. } **catch** (DoNotTerminate.ExitTrappedException e) {
20. System.out.println("Unsuccessful Termination!!");
21. }
22. }
23. }
24. *//The following class will prevent you from terminating the code using exit(0)!*
25. **class** DoNotTerminate {
26. **public** **static** **class** ExitTrappedException **extends** SecurityException {
27. **private** **static** **final** **long** serialVersionUID = 1;
28. }
29. **public** **static** **void** forbidExit() {
30. **final** SecurityManager securityManager = **new** SecurityManager() {
31. @Override
32. **public** **void** checkPermission(Permission permission) {
33. **if** (permission.getName().contains("exitVM")) {
34. **throw** **new** ExitTrappedException();
35. }
36. }
37. };
38. System.setSecurityManager(securityManager);
39. }
40. }
41. Java Substring

<https://www.hackerrank.com/challenges/java-substring/problem>



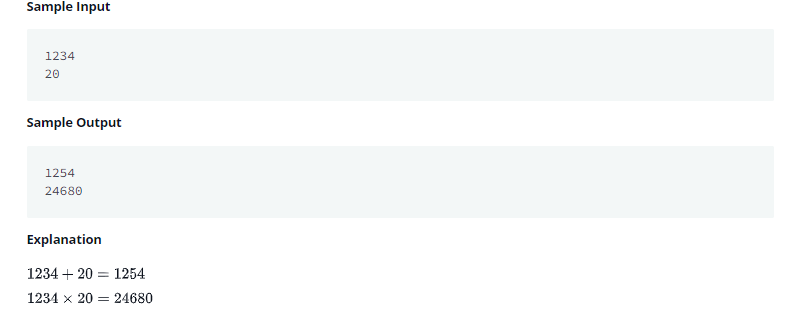
1. **import** java.io.\*;
2. **import** java.util.\*;
3. **import** java.text.\*;
4. **import** java.math.\*;
5. **import** java.util.regex.\*;
6. **public** **class** Solution {
7. **public** **static** **void** main(String[] args) {
8. Scanner in = **new** Scanner(System.in);
9. String S = in.next();
10. **int** start = in.nextInt();
11. **int** end = in.nextInt();
12. System.out.println(S.substring(start, end));
13. }
14. }
15. Java Static Initializer Block

<https://www.hackerrank.com/challenges/java-static-initializer-block/problem>



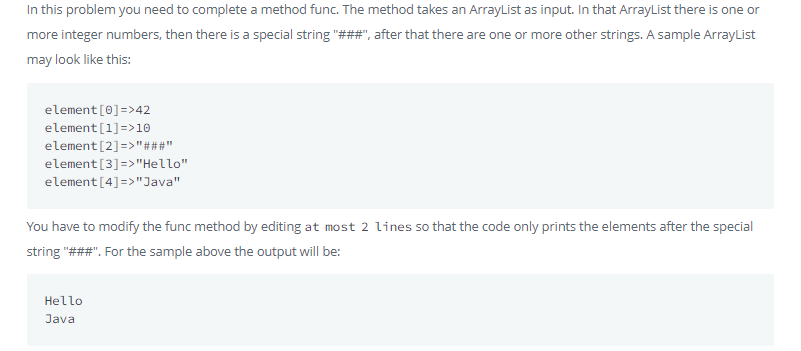
1. **import** java.io.\*;
2. **import** java.util.\*;
3. **import** java.text.\*;
4. **import** java.math.\*;
5. **import** java.util.regex.\*;
6. **public** **class** Solution {
7. *// Write your code here*
8. **private** **static** **int** B;
9. **private** **static** **int** H;
10. **private** **static** **boolean** flag;
11. **static** {
12. Scanner scan = **new** Scanner(System.in);
13. B = scan.nextInt();
14. H = scan.nextInt();
15. scan.close();
16. **if**(B <= 0 || H <= 0){
17. System.out.println("java.lang.Exception: Breadth and height must be positive");
18. flag = **false**;
19. }**else**{
20. flag = **true**;
21. }
22. }
23. **public** **static** **void** main(String[] args){
24. **if**(flag){
25. **int** area=B\*H;
26. System.out.print(area);
27. }
29. }*//end of main*
30. }*//end of class*
31. Java BigInteger

<https://www.hackerrank.com/challenges/java-biginteger/problem>



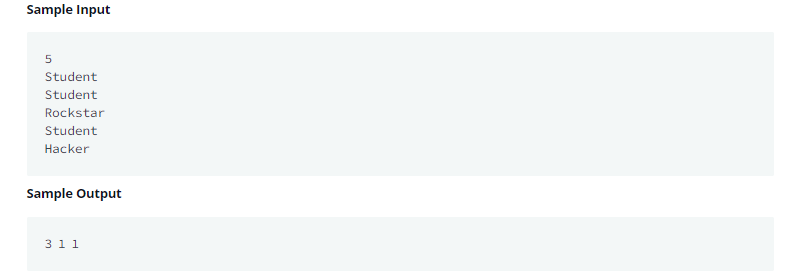
1. **import** java.io.\*;
2. **import** java.util.\*;
3. **import** java.text.\*;
4. **import** java.math.\*;
5. **import** java.util.regex.\*;
6. **public** **class** Solution {
7. **public** **static** **void** main(String[] args) {
8. */\* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. \*/*
9. Scanner scan = **new** Scanner(System.in);
10. BigInteger num1;
11. BigInteger num2;
12. BigInteger sum = BigInteger.valueOf(0);
13. BigInteger multiply = BigInteger.valueOf(0);
15. num1 = **new** BigInteger(scan.nextLine());
16. num2 = **new** BigInteger(scan.nextLine());
17. sum = sum.add(num1);
18. sum = sum.add(num2);
19. multiply = num1.multiply(num2);
20. System.out.println(sum);
21. System.out.println(multiply);
22. }
23. }
24. Java Iterator

<https://www.hackerrank.com/challenges/java-iterator/problem>



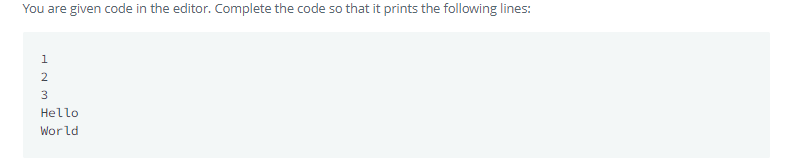
1. **import** java.util.\*;
2. **public** **class** Main{
4. **static** Iterator func(ArrayList mylist){
5. Iterator it=mylist.iterator();
6. **while**(it.hasNext()){
7. Object element = it.next();
8. **if**(element.equals("###"))*//Hints: use instanceof operator*
9. **break**;
10. }
11. **return** it;
13. }
14. @SuppressWarnings({ "unchecked" })
15. **public** **static** **void** main(String []args){
16. ArrayList mylist = **new** ArrayList();
17. Scanner sc = **new** Scanner(System.in);
18. **int** n = sc.nextInt();
19. **int** m = sc.nextInt();
20. **for**(**int** i = 0;i<n;i++){
21. mylist.add(sc.nextInt());
22. }
24. mylist.add("###");
25. **for**(**int** i=0;i<m;i++){
26. mylist.add(sc.next());
27. }
29. Iterator it=func(mylist);
30. **while**(it.hasNext()){
31. Object element = it.next();
32. System.out.println((String)element);
33. }
34. }
35. }
36. Java Instanceof keyword

<https://www.hackerrank.com/challenges/java-instanceof-keyword/problem>



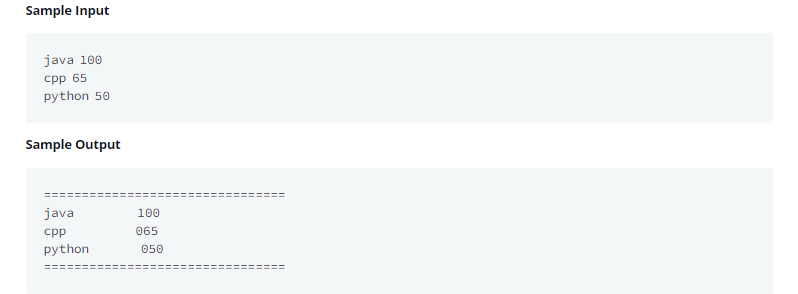
1. **import** java.util.\*;
2. **class** Student{}
3. **class** Rockstar{}
4. **class** Hacker{}
5. **public** **class** InstanceOFTutorial{
7. **static** String count(ArrayList mylist){
8. **int** a = 0,b = 0,c = 0;
9. **for**(**int** i = 0; i < mylist.size(); i++){
10. Object element=mylist.get(i);
11. **if**(element **instanceof** Student)
12. a++;
13. **if**(element **instanceof** Rockstar)
14. b++;
15. **if**(element **instanceof** Hacker)
16. c++;
17. }
18. String ret = Integer.toString(a)+" "+ Integer.toString(b)+" "+ Integer.toString(c);
19. **return** ret;
20. }
21. **public** **static** **void** main(String []args){
22. ArrayList mylist = **new** ArrayList();
23. Scanner sc = **new** Scanner(System.in);
24. **int** t = sc.nextInt();
25. **for**(**int** i=0; i<t; i++){
26. String s=sc.next();
27. **if**(s.equals("Student"))mylist.add(**new** Student());
28. **if**(s.equals("Rockstar"))mylist.add(**new** Rockstar());
29. **if**(s.equals("Hacker"))mylist.add(**new** Hacker());
30. }
31. System.out.println(count(mylist));
32. }
33. }
34. Java Generics

<https://www.hackerrank.com/challenges/java-generics/problem>



1. **import** java.io.IOException;
2. **import** java.lang.reflect.Method;
3. **class** Printer
4. {
5. *//Write your code here*
6. <T> **void** printArray(T[] array) {
7. **for** (T element : array) {
8. System.out.println(element);
9. }
10. }
11. }
12. **public** **class** Solution {
13. **public** **static** **void** main( String args[] ) {
14. Printer myPrinter = **new** Printer();
15. Integer[] intArray = { 1, 2, 3 };
16. String[] stringArray = {"Hello", "World"};
17. myPrinter.printArray(intArray);
18. myPrinter.printArray(stringArray);
19. **int** count = 0;
20. **for** (Method method : Printer.**class**.getDeclaredMethods()) {
21. String name = method.getName();
22. **if**(name.equals("printArray"))
23. count++;
24. }
25. **if**(count > 1)System.out.println("Method overloading is not allowed!");
27. }
28. }
29. Java Output Formatting

<https://www.hackerrank.com/challenges/java-output-formatting/problem>



1. **import** java.util.Scanner;
2. **public** **class** Solution {
3. **public** **static** **void** main(String[] args) {
4. Scanner sc=**new** Scanner(System.in);
5. System.out.println("================================");
6. **for**(**int** i=0;i<3;i++)
7. {
8. String s1=sc.next();
9. **int** x=sc.nextInt();
10. *//Complete this line*
11. System.out.printf("%-15s%03d\n", s1, x);
12. }
13. System.out.println("================================");
14. }
15. }

**Medium**

1. Tag Content Extractor

<https://www.hackerrank.com/challenges/tag-content-extractor/problem>



1. **import** java.io.\*;
2. **import** java.util.\*;
3. **import** java.text.\*;
4. **import** java.math.\*;
5. **import** java.util.regex.\*;
6. **public** **class** Solution{
7. **public** **static** **void** main(String[] args){
9. Scanner in = **new** Scanner(System.in);
10. **int** testCases = Integer.parseInt(in.nextLine());
11. **while**(testCases>0){
12. String line = in.nextLine();
14. *//Write your code here*
15. **boolean** matchFound = **false**;
16. Pattern r = Pattern.compile("<(.+)>([^<]+)</\\1>");
17. Matcher m = r.matcher(line);
18. **while**(m.find()){
19. System.out.println(m.group(2));
20. matchFound = **true**;
21. }**if**(!matchFound){
22. System.out.println("None");
23. }
24. testCases--;
25. }
26. }
27. }