

Task 5: Basic Number Theory 1

TIC TAC TOE

King TEGEVER OF time limit exceeded is really fascinated about Tic Tac Toe. He organizes a national level contest for Tic Tac Toe every year in Time Limit exceeded. Every year the contest has ~~lakhs~~ of participants. This year there are n participants from all over the country. The king wants to know the sum of $a_i \oplus i$ from 1 to n . Now as you already know Mohan is not good with maths, he asks you to help him.

Aim: To write and execute the program for given scenario based on basic number theory - 1

Algorithm:

1. Read the integer n .
2. calculate the minimum value of the sum as $((n*(n-1)/2) \oplus n)$ modulo $10^9 + 7$
3. calculate the maximum value of the sum as $((((n*(n-1)/2) * (2*n-1))/3) \oplus n)$ modulo $10^9 + 7$
4. Print the minimum and maximum value of the sum as output.
5. end the program.

I - profit statement

Output:

No. of inputs: 20 hours fixed cost to INPUTS

5 factors fixed labour & machine etc. cost per unit

20 30

Cost labour fixed cost per hour

cost of labour per hour

fixed cost of labour per hour

cost of labour per hour

where this book form is normal cost formula way is

method of way also

normal cost margin per hour labour cost of input

I - profit statement based on labour cost

method A

respecting all book

(C - V%) x m/s + to cover production all overheads

FOB delivery

(C - V%) x m/s + to cover production all overheads

FOB delivery $\Rightarrow (C - V%) \times (S - (C - V)M)$

m/s to cover production and manufacturing all fixed

expenses

method of book

Ran

Program:

```
#include <stdio.h>
#include <math.h>
#define MOD 1000000007
int main()
{
    int t;
    scanf("%d", &t);
    while(t--) {
        long long n;
        scanf("%lld", &n);
        long long min_sum = ((n-1)*(n-1)*(n/4)) % MOD;
        long long max_sum = (((n+1)*n*(n+1)*(3*(n+1)/12)) % MOD);
        printf("%lld\n", min_sum, max_sum);
    }
    return 0;
}
```

problem:

Given two integers, and a recursive technique to find their GCD is the Euclidean Algorithm. The algorithm states that, for computing the GCD of two positive integers if and are equal there are few optimizations that can be made to the above logic.

Output:

sample Input:

1 5

sample Output:

1

Algorithm

1. Start

2. Read n and m

3. Initialize $s = 1$

4. Initialize $c = 0$

5. Initialize $d = 0$

6. Initialize $r = n$

7. Initialize $t = m$

8. Initialize $g = 1$

9. Initialize $h = 1$

10. Initialize $i = 1$

11. Initialize $j = 1$

12. Initialize $k = 1$

13. Initialize $l = 1$

14. Initialize $m = 1$

15. Initialize $p = 1$

16. Initialize $q = 1$

17. Initialize $r = 1$

18. Initialize $s = 1$

19. Initialize $t = 1$

20. Initialize $u = 1$

21. Initialize $v = 1$

22. Initialize $w = 1$

23. Initialize $x = 1$

24. Initialize $y = 1$

25. Initialize $z = 1$

26. Initialize $aa = 1$

27. Initialize $ab = 1$

28. Initialize $ac = 1$

29. Initialize $ad = 1$

30. Initialize $ba = 1$

31. Initialize $bb = 1$

32. Initialize $bc = 1$

33. Initialize $bd = 1$

34. Initialize $ca = 1$

35. Initialize $cb = 1$

36. Initialize $cc = 1$

37. Initialize $cd = 1$

38. Initialize $da = 1$

39. Initialize $db = 1$

40. Initialize $dc = 1$

41. Initialize $dd = 1$

42. Initialize $ea = 1$

43. Initialize $eb = 1$

44. Initialize $ec = 1$

45. Initialize $ed = 1$

46. Initialize $fa = 1$

47. Initialize $fb = 1$

48. Initialize $fc = 1$

49. Initialize $fd = 1$

50. Initialize $ga = 1$

51. Initialize $gb = 1$

52. Initialize $gc = 1$

53. Initialize $gd = 1$

54. Initialize $ha = 1$

55. Initialize $hb = 1$

56. Initialize $hc = 1$

57. Initialize $hd = 1$

58. Initialize $ia = 1$

59. Initialize $ib = 1$

60. Initialize $ic = 1$

61. Initialize $id = 1$

62. Initialize $ja = 1$

63. Initialize $jb = 1$

64. Initialize $jc = 1$

65. Initialize $jd = 1$

66. Initialize $ka = 1$

67. Initialize $kb = 1$

68. Initialize $kc = 1$

69. Initialize $kd = 1$

70. Initialize $la = 1$

71. Initialize $lb = 1$

72. Initialize $lc = 1$

73. Initialize $ld = 1$

74. Initialize $ma = 1$

75. Initialize $mb = 1$

76. Initialize $mc = 1$

77. Initialize $md = 1$

78. Initialize $na = 1$

79. Initialize $nb = 1$

80. Initialize $nc = 1$

81. Initialize $nd = 1$

82. Initialize $ea = 1$

83. Initialize $eb = 1$

84. Initialize $ec = 1$

85. Initialize $ed = 1$

86. Initialize $fa = 1$

87. Initialize $fb = 1$

88. Initialize $fc = 1$

89. Initialize $fd = 1$

90. Initialize $ga = 1$

91. Initialize $gb = 1$

92. Initialize $gc = 1$

93. Initialize $gd = 1$

94. Initialize $ha = 1$

95. Initialize $hb = 1$

96. Initialize $hc = 1$

97. Initialize $hd = 1$

98. Initialize $ia = 1$

99. Initialize $ib = 1$

100. Initialize $ic = 1$

101. Initialize $id = 1$

102. Initialize $ja = 1$

103. Initialize $jb = 1$

104. Initialize $jc = 1$

105. Initialize $jd = 1$

106. Initialize $ka = 1$

107. Initialize $kb = 1$

108. Initialize $kc = 1$

109. Initialize $kd = 1$

110. Initialize $la = 1$

111. Initialize $lb = 1$

112. Initialize $lc = 1$

113. Initialize $ld = 1$

114. Initialize $na = 1$

115. Initialize $nb = 1$

116. Initialize $nc = 1$

117. Initialize $nd = 1$

118. Initialize $ea = 1$

119. Initialize $eb = 1$

120. Initialize $ec = 1$

121. Initialize $ed = 1$

122. Initialize $fa = 1$

123. Initialize $fb = 1$

124. Initialize $fc = 1$

125. Initialize $fd = 1$

126. Initialize $ga = 1$

127. Initialize $gb = 1$

128. Initialize $gc = 1$

129. Initialize $gd = 1$

130. Initialize $ha = 1$

131. Initialize $hb = 1$

132. Initialize $hc = 1$

133. Initialize $hd = 1$

134. Initialize $ia = 1$

135. Initialize $ib = 1$

136. Initialize $ic = 1$

137. Initialize $id = 1$

138. Initialize $ja = 1$

139. Initialize $jb = 1$

140. Initialize $jc = 1$

141. Initialize $jd = 1$

142. Initialize $ka = 1$

143. Initialize $kb = 1$

144. Initialize $kc = 1$

145. Initialize $kd = 1$

146. Initialize $la = 1$

147. Initialize $lb = 1$

148. Initialize $lc = 1$

149. Initialize $ld = 1$

150. Initialize $na = 1$

151. Initialize $nb = 1$

152. Initialize $nc = 1$

153. Initialize $nd = 1$

154. Initialize $ea = 1$

155. Initialize $eb = 1$

156. Initialize $ec = 1$

157. Initialize $ed = 1$

158. Initialize $fa = 1$

159. Initialize $fb = 1$

160. Initialize $fc = 1$

161. Initialize $fd = 1$

162. Initialize $ga = 1$

163. Initialize $gb = 1$

164. Initialize $gc = 1$

165. Initialize $gd = 1$

166. Initialize $ha = 1$

167. Initialize $hb = 1$

168. Initialize $hc = 1$

169. Initialize $hd = 1$

170. Initialize $ia = 1$

171. Initialize $ib = 1$

172. Initialize $ic = 1$

173. Initialize $id = 1$

174. Initialize $ja = 1$

175. Initialize $jb = 1$

176. Initialize $jc = 1$

177. Initialize $jd = 1$

178. Initialize $ka = 1$

179. Initialize $kb = 1$

180. Initialize $kc = 1$

181. Initialize $kd = 1$

182. Initialize $la = 1$

183. Initialize $lb = 1$

184. Initialize $lc = 1$

185. Initialize $ld = 1$

186. Initialize $na = 1$

187. Initialize $nb = 1$

188. Initialize $nc = 1$

189. Initialize $nd = 1$

190. Initialize $ea = 1$

191. Initialize $eb = 1$

192. Initialize $ec = 1$

193. Initialize $ed = 1$

194. Initialize $fa = 1$

195. Initialize $fb = 1$

196. Initialize $fc = 1$

197. Initialize $fd = 1$

198. Initialize $ga = 1$

199. Initialize $gb = 1$

200. Initialize $gc = 1$

201. Initialize $gd = 1$

202. Initialize $ha = 1$

203. Initialize $hb = 1$

204. Initialize $hc = 1$

205. Initialize $hd = 1$

206. Initialize $ia = 1$

207. Initialize $ib = 1$

208. Initialize $ic = 1$

209. Initialize $id = 1$

210. Initialize $ja = 1$

211. Initialize $jb = 1$

212. Initialize $jc = 1$

213. Initialize $jd = 1$

214. Initialize $ka = 1$

215. Initialize $kb = 1$

216. Initialize $kc = 1$

217. Initialize $kd = 1$

218. Initialize $la = 1$

219. Initialize $lb = 1$

220. Initialize $lc = 1$

221. Initialize $ld = 1$

222. Initialize $na = 1$

223. Initialize $nb = 1$

224. Initialize $nc = 1$

225. Initialize $nd = 1$

226. Initialize $ea = 1$

227. Initialize $eb = 1$

228. Initialize $ec = 1$

Algorithm:

1. Read the two positive integers a and b .
2. If b is zero, return a as the GCD.
3. Otherwise, recursively call the function with arguments b and the remainder of a divided by b .
4. Return the result of the recursive call as the GCD.

Program:

```
#include <stdio.h>
int gcd(int a, int b) {
    if (b == 0) {
        return a;
    }
    return gcd(b, a % b);
}
int main() {
    int a, b;
    scanf("%d %d", &a, &b);
    int result = gcd(a, b);
    printf("%d\n", result);
    return 0;
}
```

Result: The program is executed and verified successfully.

VEL TECH - CSE	
PG NO.	8
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	7
RECORD (4)	4
TOTAL (15)	19
SIGN WITH DATE	