Date : Page No :

Assignment Number

Problem Statement

Program in C to multiply two integers using russian peasant method.

Theory

The russian peasant method is used to calculate the result of multiplication of two integer numbers by the process of successive multiplication by 2 and addition. It is a systematic method for multiplying two numbers that does not require the multiplication table, only the ability to multiply and divide by 2, and to add. It decomposes one of the multiplicands (preferably the smaller) into a sum of powers of two and creates a table of doublings of the second multiplicand. This method may be called **mediation and duplation**, where mediation means halving one number and duplation means doubling the other number.

Example : Multiplication of 15 and 16 using the above method will be as following :

Right Column	Left column
15	16
7	32
3	64
1	128

Result : 16+32+64+128 = 240

Algorithm

Input : Two numbers to multiply, say a and b.
Output : The result of multiplication of the two variables, say s.
Steps :

Step 1. Set loop = True
Step 2. Repeat through step 2.A to 2.Q while(loop = True)
A) Print "Enter two integers : "
B) Input a, b
C) If(a < 1 Or b < 1)
a) Print "Invalid input!”
b) End
[End of if structure]
D) Print "Peasant Multiplication Method as follows”
E) Print "=="
F) Set i = 1
G) Set d = a
H) Set e = b
I) Repeat through step I.a to I.d 2hile(d > 0)
a) If(d mod 2 = 0)
i. Print d " " e " x(struck of)"
b) Else
i. Print d " " e
ii. Set n[i] = e
iii. Set i = i + 1
[End of if structure]
c) Set d = d/2
d) Set e = e*2
[End of while loop]
J) Set s = n[1]
K) Print s
L) Set d = 2
M) Repeat through step M.a to M.c while(d < i)
a) Print " + " n[d]

b) Set s = s+n[d]
c) Set d = d+1
[End of while loop]
N) Print " = " s
O) Print " which is equal to " a " * " b " = " a*b
P) Print "Do you want to continue (y/n) ? "
Q) Input choice
R) If(choice != "Y" And choice != "y")
a) Set loop = False
[End of if structure]
[End of while loop]

Source Code

```
#include <stdio.h>
int main(){
	int a, b, d, p, n[20], s, i;
	char ch = 'Y';
	do{
		printf("\nEnter the two numbers : ");
		scanf("%d%d", &a, &b);
		printf("\nPeasant Multiplication Method as follows : ");
		printf("\n==================================”);
		i = 0;
		for(d = a, p = b;d > 0;d = d/2, p = p*2){
			if(d % 2 == 0)
				printf("\n\n%3d\t\t%3d x(struck of)", d, p);
			else{
				printf("\n\n%3d\t\t%3d", d, p);
				n[i++] = p;
			}
		}
		s = n[0];
		printf("\n\n %d", n[0]);
		for(d = 1;d < i;d++){
			s = s + n[d];
			printf(" + %d", n[d]);
		}
		printf(" = %d", s);
		printf(" which is equal to %d * %d = %d", a, b, (a*b));
		printf("\nDo you want to continue (y/n) ? ");
		scanf(" %c", &ch);
	} while(ch == 'Y' || ch == 'y');
	return 0; }
```

Input and Output

Set 1

Enter the two numbers : 15 17

Peasant Multiplication Method as follows :
==========
15 17

7 34

3 68

1 136

17 + 34 + 68 + 136 = 255 which is equal to 15 * 17 = 255
Do you want to continue (y/n) ? n

Set 2

Enter the two numbers : 10 14

Peasant Multiplication Method as follows :
==========
10 14 x(struck of)

5 28

2 56 x(struck of)

1 112

28 + 112 = 140 which is equal to 10 * 14 = 140
Do you want to continue (y/n) ? n

Discussion

1. This algorithm does not support multiplication of negative numbers.
2. The advantages of this algorithm include easy learning, no memorization required and can be performed by tokens if pen and paper are not available.
3. It does take more number of steps than normal multiplication for long numbers, hence it can be unwieldy when large numbers are involved.