|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| BOOTHS ALGORITHM  PROGRAM:   |  |  |  | | --- | --- | --- | |  |  | #include <math.h> | |  |  |  | |  |  | int a = 0,b = 0, c = 0, a1 = 0, b1 = 0, com[5] = { 1, 0, 0, 0, 0}; | |  |  | int anum[5] = {0}, anumcp[5] = {0}, bnum[5] = {0}; | |  |  | int acomp[5] = {0}, bcomp[5] = {0}, pro[5] = {0}, res[5] = {0}; | |  |  |  | |  |  | void binary(){ | |  |  | a1 = fabs(a); | |  |  | b1 = fabs(b); | |  |  | int r, r2, i, temp; | |  |  | for (i = 0; i < 5; i++){ | |  |  | r = a1 % 2; | |  |  | a1 = a1 / 2; | |  |  | r2 = b1 % 2; | |  |  | b1 = b1 / 2; | |  |  | anum[i] = r; | |  |  | anumcp[i] = r; | |  |  | bnum[i] = r2; | |  |  | if(r2 == 0){ | |  |  | bcomp[i] = 1; | |  |  | } | |  |  | if(r == 0){ | |  |  | acomp[i] =1; | |  |  | } | |  |  | } | |  |  | //part for two's complementing | |  |  | c = 0; | |  |  | for ( i = 0; i < 5; i++){ | |  |  | res[i] = com[i]+ bcomp[i] + c; | |  |  | if(res[i] >= 2){ | |  |  | c = 1; | |  |  | } | |  |  | else | |  |  | c = 0; | |  |  | res[i] = res[i] % 2; | |  |  | } | |  |  | for (i = 4; i >= 0; i--){ | |  |  | bcomp[i] = res[i]; | |  |  | } | |  |  | //in case of negative inputs | |  |  | if (a < 0){ | |  |  | c = 0; | |  |  | for (i = 4; i >= 0; i--){ | |  |  | res[i] = 0; | |  |  | } | |  |  | for ( i = 0; i < 5; i++){ | |  |  | res[i] = com[i] + acomp[i] + c; | |  |  | if (res[i] >= 2){ | |  |  | c = 1; | |  |  | } | |  |  | else | |  |  | c = 0; | |  |  | res[i] = res[i]%2; | |  |  | } | |  |  | for (i = 4; i >= 0; i--){ | |  |  | anum[i] = res[i]; | |  |  | anumcp[i] = res[i]; | |  |  | } | |  |  |  | |  |  | } | |  |  | if(b < 0){ | |  |  | for (i = 0; i < 5; i++){ | |  |  | temp = bnum[i]; | |  |  | bnum[i] = bcomp[i]; | |  |  | bcomp[i] = temp; | |  |  | } | |  |  | } | |  |  | } | |  |  | void add(int num[]){ | |  |  | int i; | |  |  | c = 0; | |  |  | for ( i = 0; i < 5; i++){ | |  |  | res[i] = pro[i] + num[i] + c; | |  |  | if (res[i] >= 2){ | |  |  | c = 1; | |  |  | } | |  |  | else{ | |  |  | c = 0; | |  |  | } | |  |  | res[i] = res[i]%2; | |  |  | } | |  |  | for (i = 4; i >= 0; i--){ | |  |  | pro[i] = res[i]; | |  |  | printf("%d",pro[i]); | |  |  | } | |  |  | printf(":"); | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d", anumcp[i]); | |  |  | } | |  |  | } | |  |  | void arshift(){//for arithmetic shift right | |  |  | int temp = pro[4], temp2 = pro[0], i; | |  |  | for (i = 1; i < 5 ; i++){//shift the MSB of product | |  |  | pro[i-1] = pro[i]; | |  |  | } | |  |  | pro[4] = temp; | |  |  | for (i = 1; i < 5 ; i++){//shift the LSB of product | |  |  | anumcp[i-1] = anumcp[i]; | |  |  | } | |  |  | anumcp[4] = temp2; | |  |  | printf("\nAR-SHIFT: ");//display together | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d",pro[i]); | |  |  | } | |  |  | printf(":"); | |  |  | for(i = 4; i >= 0; i--){ | |  |  | printf("%d", anumcp[i]); | |  |  | } | |  |  | } | |  |  |  | |  |  | int main(){ | |  |  | int i, q = 0; | |  |  | printf("\t\tBOOTH'S MULTIPLICATION ALGORITHM"); | |  |  | printf("\nEnter two numbers to multiply: "); | |  |  | printf("\nBoth must be less than 16"); | |  |  | //simulating for two numbers each below 16 | |  |  | do{ | |  |  | printf("\nEnter A: "); | |  |  | scanf("%d",&a); | |  |  | printf("Enter B: "); | |  |  | scanf("%d", &b); | |  |  | }while(a >=16 || b >=16); | |  |  |  | |  |  | printf("\nExpected product = %d", a \* b); | |  |  | binary(); | |  |  | printf("\n\nBinary Equivalents are: "); | |  |  | printf("\nA = "); | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d", anum[i]); | |  |  | } | |  |  | printf("\nB = "); | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d", bnum[i]); | |  |  | } | |  |  | printf("\nB'+ 1 = "); | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d", bcomp[i]); | |  |  | } | |  |  | printf("\n\n"); | |  |  | for (i = 0;i < 5; i++){ | |  |  | if (anum[i] == q){//just shift for 00 or 11 | |  |  | printf("\n-->"); | |  |  | arshift(); | |  |  | q = anum[i]; | |  |  | } | |  |  | else if(anum[i] == 1 && q == 0){//subtract and shift for 10 | |  |  | printf("\n-->"); | |  |  | printf("\nSUB B: "); | |  |  | add(bcomp);//add two's complement to implement subtraction | |  |  | arshift(); | |  |  | q = anum[i]; | |  |  | } | |  |  | else{//add ans shift for 01 | |  |  | printf("\n-->"); | |  |  | printf("\nADD B: "); | |  |  | add(bnum); | |  |  | arshift(); | |  |  | q = anum[i]; | |  |  | } | |  |  | } | |  |  |  | |  |  | printf("\nProduct is = "); | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d", pro[i]); | |  |  | } | |  |  | for (i = 4; i >= 0; i--){ | |  |  | printf("%d", anumcp[i]); | |  |  | } | |  |  | } |   OUTPUT:  Enter two numbers to multiply:  Both must be less than 16  Enter A: 2  Enter B: 3  Expected product = 6  Binary Equivalents are:  A = 00010  B = 00011  B'+ 1 = 11101  -->  AR-SHIFT: 00000:00001  -->  SUB B: 11101:00001  AR-SHIFT: 11110:10000  -->  ADD B: 00001:10000  AR-SHIFT: 00000:11000  -->  AR-SHIFT: 00000:01100  -->  AR-SHIFT: 00000:00110  Product is = 0000000110  --------------------------------  Process exited after 1.927 seconds with return value 1  Press any key to continue . . . |