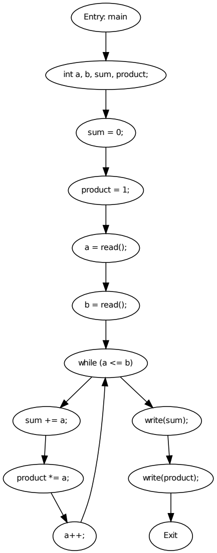
Exercise Sheet 10

Exercise 1.1: Effects of statements

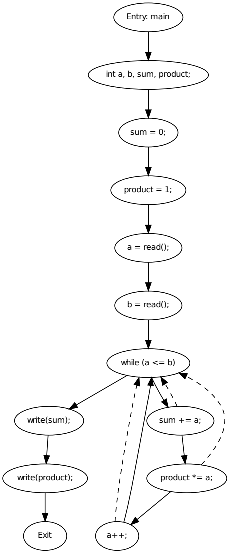
statement	read	write
`int a, b, sum, product;`	-	a, b, sum, product
`sum = 0;`	-	sum
`product = 1;`	-	product
`a = read();`	-	a
`b = read();`	-	b
`while (a <= b)`	a, b	-
`sum += a;`	sum, a	sum
`product *= a;`	product, a	product
`a++;`	a	a
`write(sum);`	sum	-
`write(product);`	product	-

Exercise 1.2: Control-Flow-Graph

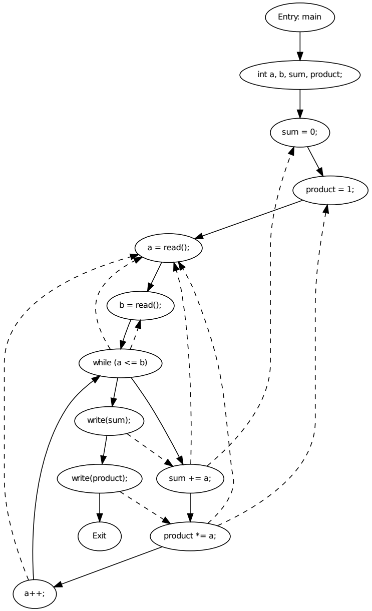
```
void main()
{
    int a, b, sum, product;
    sum = 0;
    product = 1;
    a = read();
    b = read();
    while (a <= b)
    {
        sum += a;
        product *= a;
        a++;
    }
    write(sum);
    write(product);
}
```



Exercise 1.3: Control Dependencies



Exercise 1.4: Data Dependencies



Exercise 2: Fixing defects

The dd_{min} algorithm will return the string “z”. The reason is the wrong initialization of the array `cnt`. In particular, its length should we equal to the number of letters in the English alphabet, i.e., the appropriate size would be 26 elements, whereas the program allocates memory only for 25 elements and thus no memory is allocated to count the number of occurances of the letter “z”.

We can locate this defect by first looking at the line 19 (`cnt[s.charAt(i)a]++;`) where the exception is thrown. This line is data dependent on the line 13 (`int[] cnt = new int[25];`). Now we observe that the array `cnt` is first declared there and thus the defect originates at this line of code.