

//Calculate the product of two natural numbers

LOOP:

POB PRODUCT

DOD A

LAD PRODUCT

POB B

ODE ONE

SOZ END

LAD B

SOB LOOP

END:

POB PRODUCT

STP

ONE: RST 1

A: RST 6

B: RST 3

PRODUCT: RST 0

//Search an array for a value specified in a separate variable and store the index of its first occurrence.

//use an additional variable to indicate the end of an array

//CharValue: RST 0 // additional variable

//ARRAY: RST 1

//RST 2

//RST 3

//RST 0 // end of an array

NEXT: POB ARRAY

SOM END_OF_ARRAY

ODE SEARCHED

SOZ FOUND

L1:

POB INDEX

DOD ONE

LAD INDEX

POB NEXT

DOD ONE

LAD NEXT

SOB NEXT

FOUND: STP

END_OF_ARRAY: STP

SEARCHED: RST 7

INDEX: RST 1

ONE: RST 1

CharValue: RST -1 // additional variable

ARRAY: RST 1

RST 2

RST 2

RST 7

RST -1 // end of an array

//Decrement all elements of an array.

loop:

POB size

ODE one

SOM end

LAD SIZE

inst1: POB ARRAY

ODE DecrementBy

inst2: LAD ARRAY

POB inst1

DOD one

LAD inst1

POB inst2

DOD one

LAD inst2

SOB loop

end:

STP

SIZE: RST 8 //number of elements: 3

ARRAY: RST 8

RST 10

RST 3

RST 1

RST 2

RST 3

RST 4

RST 5

DecrementBy: RST 2

one: RST 1

//Search an array for a value specified in a separate variable and count the number of occurrences.

loop:

inst: POB array

ODE CharValue

SOZ end

DOD CharValue

//checking the value in the array

ODE LookedForValue

SOZ OccurenceFound

SOB continue

OccurenceFound: POB occurrences

DOD one

LAD occurrences

continue: POB inst

DOD one

LAD inst

SOB loop

end: POB occurrences

STP

CharValue: RST 0

ARRAY: RST -3

RST 2

RST 3

RST 5

RST 6

RST 5

RST 10

RST -3

RST 0

LookedForValue: RST -3

occurences: RST 0

one: RST 1

//Decrement all elements of an array.

//the number of elements of the array is specified in a separate variable

//SIZE: RST 3 //number of elements: 3

//ARRAY: RST 1

//RST 2

//RST 3

TABLE_SIZE: POB SIZE

ODE ONE

SOM END

LAD SIZE

SUB: POB ARRAY

ODE ONE

WHERE: LAD ARRAY

L1: POB SUB

DOD ONE

LAD SUB

POB WHERE

DOD ONE

LAD WHERE

SOB TABLE_SIZE

END: STP

ONE: RST 1

SIZE: RST 2

ARRAY: RST 1

RST 2

RST 3

RST 4

RST 5

RST 6

//product of 2 natural numbers

//SDP Product

//STP

Main:

//determining the counter value

POB N2

SOZ end

SOM Negative

LAD Counter

//determining the Product value

POB N1

SOZ end

LAD Product

LAD First_Prdct

loop:

POB Counter

ODE One

LAD Counter

SOZ end

POB Product

DOD First_Prdct

LAD Product

SOB loop

//if counter is negative, change values of counter and Product

Negative:

LAD Product

LAD First_Prduct

POB N1

SOZ end

LAD Counter

SOB loop

end:

POB Product

STP

One: RST 1

N1: RST 4

N2: RST 10

First_Prduct: RPA

Counter: RPA

Product: RPA

//adding all elements in the array

loop:

POB size

ODE one

SOM end

LAD size

POB sum

inst: DOD array

LAD sum

//incrementing index of the array

POB inst

DOD one

LAD inst

SOB loop

end: POB sum

STP

array: RST 1

RST 3

RST 4

RST 3

size: RST 4

one: RST 1

sum: RST 0

//Evaluate exponentiation of two natural numbers

POB Final

DOD Num1

LAD Final

POB Num2

SOZ end_exp_one

POB Num1

SOZ end_exp_zero

Main_exp:

POB Num2

ODE one

SOZ end_exp

LAD Num2

//Program below multiplies 2 numbers

Main_product:

POB Num1

LAD N1

POB Final

LAD N2

//determining the counter value

POB N2

SOZ end

SOM Negative

LAD Counter

//determining the Product value

POB N1

SOZ end

LAD Product

LAD First_Prdct

loop:

POB Counter

ODE One

LAD Counter

SOZ end

POB Product

DOD First_Prdct

LAD Product

SOB loop

//if counter is negative, change values of counter and Product

Negative:

LAD Product

LAD First_Prdct

POB N1

SOZ end

LAD Counter

SOB loop

end:

POB Product

LAD Final

SOB Main_exp

//End of multiplication

end_exp:

POB Final

STP

end_exp_one:

POB one

STP

end_exp_zero:

POB zero

STP

One: RST 1

N1: RST 0

N2: RST 0

First_Prduct: RPA

Counter: RPA

Product: RPA

//For multiplication

Num1: RST 5

Num2: RST 0

Final: RST 0

zero: RST 0