Main, BSA GetOperator

BSA GetSignedInt //

STA Num

BSA GetSignedInt

STA Num2

BUN FI

//Main Data

Num, DEC 0

Num2, DEC 0

Res, DEC 0

//

GetOperator, HEX 0 // HERE IT ALL STARTS

CLA

BSA In\_char

CheckMul, CMA // IF One of the operators == zero then

INC

ADD Mul

SZA

BUN CheckPlus

STA Mul

BUN GetOperator I

CheckPlus, LDA TOperator

CMA

INC

ADD Plus

SZA

BUN CheckMinus

STA Plus

BUN GetOperator I

CheckMinus, LDA TOperator

CMA

INC

ADD Minus

SZA

BUN CheckDiv

STA Minus

BUN GetOperator I

CheckDiv, LDA TOperator

CMA

INC

ADD Div

SZA

BUN FI

STA Div

BUN GetOperator I

GetSignedInt, HEX 0

CLA

STA TNum

LDA FlagOFF

STA minus\_Flag

In\_char, HEX 0

BSA Getc

STA Cc

LDA operatorFlag

SZA

BUN MinusMinus //IF OPERATOR START

LDA Cc

STA TOperator

LDA operatorFlag

INC

STA operatorFlag

LDA TOperator

BUN In\_char I

MinusMinus, LDA Cc

ADD minusMinusASCII

SZA

BUN operatorFlagCheck

LDA FlagOn //FlagOn starts with negative number

STA minus\_Flag

BSA Getc

STA Cc

operatorFlagCheck, LDA OperatorFlag

SZA

BUN getUnsigned //if operator phase is already executed

BUN GetSignedInt I

getUnsigned, LDA Cc

ADD minusCReturn

SZA

BUN Convert

BUN End\_loop

Convert, LDA Cc // IF Character is not ENTER or Operator

ADD mASCII\_zero //CONVERT TO the real number - 30

STA Cc

DigitCase, LDA TNum // The current whole number we have

BSA MultBy10

ADD Cc // Cc is currently the original next number

STA TNum // TNum = (prev digit\*10) + currentDigit

BUN In\_char

End\_loop, LDA minus\_Flag

SPA

BUN FI\_1

LDA TNum

CMA

INC

STA TNum

FI\_1, LDA TNum

BUN GetSignedInt I

FI, HEX 0

SENDTOPLUS, LDA Plus

SZA

BUN SENDTOMINUS

LDA Num

ADD Num2

STA Res

BUN PRINTRES

SENDTOMINUS, HLT

PRINTRES, ADD mASCII\_zero

OUT

HLT

//

// GetSignedInt Data

minusMinusASCII, HEX -2D

OperatorFlag, DEC 0

minus\_Flag, DEC 0

FlagOn, DEC 1

FlagOff, DEC 0

// GetOperatorData

TOperator, DEC 0

Mul, HEX 2A

Minus, HEX 2D

Plus, HEX 2B

Div, HEX 2F

ProtoMul, HEX 2A // Prototype will reassign \*/+- to the original.

ProtoMinus, HEX 2D

ProtoPlus, HEX 2B

ProtoDiv, HEX 2F

mASCII\_CONVERT, HEX 30

//GetUnsignedInt Data

TNum, DEC 0

Cc, DEC 0

mASCII\_zero, HEX -30

minusCReturn, HEX -D

MultBy10, HEX 0

CLE

CIL

STA tmp

CIL

CIL

ADD tmp

BUN MultBy10 I

// MultBy10 data

tmp, DEC 0

// subroutine getC()

Getc, HEX 0

Inp\_char, SKI

BUN Inp\_char

INP

Echo, SKO

BUN Echo

OUT

BUN Getc I

// Stack operations Push & Pop

POP, HEX 0 // AC POP() {

LDA SPtr // SPtr--;

ADD Minus1 //

STA SPtr //

LDA SPtr I // AC = \*SPtr;

BUN POP I // return } //

PUSH, HEX 0 // PUSH(AC) {

STA SPtr I // \*SPtr = AC;

ISZ SPtr // SPtr++;

BUN PUSH I

Minus1, DEC -1

SPtr, HEX 400

STACKRES, DEC 0 // THIS STACK WILL STORE RESULT

DEC 0

DEC 0

DEC 0

DEC 0

STACKTOP, DEC 0

END

-------------------------------------------------------------------

LDA Res

BSA putSignedInt // PutUnsignedInt(Num);

HLT //

Num, DEC -32768 //

putSignedInt, HEX 0 // putSignedInt(short signed int Tnum) {

STA Tnum //

LDA Power10Ptr //

STA Ptr // Ptr = @Power10Array

LDA It\_count

STA Count // Count = It\_count; i.e. 4 in the case of 16 bits

LDA TNum // IF (TNum 0; Count--)

SPA //

BUN EndFor //

CLA // DO digit =0;

STA Digit //

Loop, LDA ptr I // WHILE (Stripped - \*Ptr > 0)

ADD Stripped // DO

SNA

//

BUN Continue //

BUN Outx //

Continue, STA Stripped // Stripped=Stripped - \*Ptr;

ISZ digit // digit++;

BUN Loop // OD;

Outx, LDA digit //

ADD ascii\_Offset //

BSA Putc //

ISZ ptr // ptr++;

LDA count // Count--;

ADD Minus1 //

STA Count //

BUN ForLoop // OD;

EndFor, LDA Stripped //

ADD ascii\_Offset //

BSA Putc // Output(last digit);

// print units – the left over in ACC

End, BUN putSignedInt I // return; } //

// putSignedInt data

//TNum , DEC 0 //Temp for output digit, DEC 0 Sign\_ascii, HEX 2D // the "-" character Note: -32768 ≤ number ≤ +32767 So we can use SNA

ascii\_Offset, HEX 30 // digit to ascii representation offset Stripped, DEC 0 // +ve value of TNum //

Minus1, DEC -1

count, DEC 0

It\_count , DEC 4 // loop count (for 16 bit integer)

Power10Ptr, HEX 100 // @Power10Array

ptr, HEX 0 // //

ORG 100

Power10Array, DEC -10000 // -10 to power of 4

DEC -1000 // -10 to power of 3

DEC -100 // -10 to power of 2

DEC -10 // -10 to power of 1

//Subroutine to print a char to screen

Putc, HEX 0 // void Putc(char) {

Out, SKO

BUN Out

OUT // print(char);

BUN Putc I // }

END