Given the following grammar:

A → id = E
E → E+T | E→ E-T | T
T → T*F | T/F | F
F → (E) | id
id → a | b | c … z etc

Use bottom up parsing algorithm for the following input strings:

a) a=b+c-d
b) a=b*c+d*e
c) a=b*c*d
d) a=a/b-c*d
e) a=b+c-d-f

Answer:

a) a=b+c-d

Right Most Derivation

A → **id = E**
→ id = **E - T**
→ id = E - **F**
→ id = E - **id**
→ id = E - **d**
→ id = **E + T** - d
→ id = E + **F** - d
→ id = E + **id** - d
→ id = E + **c** - d
→ id = **T** + c - d
→ id = **F** + c - d
→ id = **id** + c - d
→ id = **b** + c - d
→ **a** = b + c - d

Handle sequence:

a → b → id →F → T → c → id → F → E+T → d → id → F → E-T → id=E

Stack	Input	Action
$	a=b+c-d$	Shift
$a	=b+c-d$	Reduce by id → a
$id	=b+c-d$	Shift
$id=	b+c-d$	Shift
$id=b	+c-d$	Reduce by id → b
$id=id	+c-d$	Reduce by F → id
$id=F	+c-d$	Reduce by T → F
$id=T	+c-d$	Reduce by E → T
$id=E	+c-d$	Shift
$id=E+	c-d$	Shift
$id=E+c	-d$	Reduce by id → c
$id=E+id	-d$	Reduce by F → id
$id=E+F	-d$	Reduce by T → F
$id=E+T	-d$	Reduce by E → E+T
$id=E	-d$	Shift
$id=E-	d$	Shift
$id=E-d	$	Reduce by id → d
$id=E-id	$	Reduce by F → id
$id=E-F	$	Reduce by T → F
$id=E-T	$	Reduce by E → E-T
$id=E	$	A accept

b) a=b*c+d*e

Right Most Derivation

A → **id = E**
→ id = **E + T**
→ id = E + **T * F**
→ id = E + T * **id**
→ id = E + T * **e**
→ id = E + **F** * e
→ id = E + **id** * e
→ id = E + **d** * e
→ id = **T * F** + d * e
→ id = T * **id** + d * e
→ id = T * **c** + d * e
→ id = **F** * c + d * e
→ id = **id** * c + d * e

→ id = **b** * c + d * e
→ **a** = b * c + d * e

Handle Sequence:
a → b → id → F → c → id → T*F → d → id → F → e → id → T*F → E+T → id=E

Stack	**Input**	**Action**
$	a=b*c+d*e$	Shift
$a	=b*c+d*e$	Reduce by id → a
$id	=b*c+d*e$	Shift
$id=	b*c+d*e$	Shift
$id=b	*c+d*e$	Reduce by id → b
$id=id	*c+d*e$	Reduce by F → id
$id=F	*c+d*e$	Reduce by T → F
$id=T	*c+d*e$	Shift
$id=T*	c+d*e$	Shift
$id=T*c	+d*e$	Reduce by id → c
$id=T*id	+d*e$	Reduce by F → id
$id=T*F	+d*e$	Reduce by E → T*F
$id=E	+d*e$	Shift
$id=E+	d*e$	Shift
$id=E+d	*e$	Reduce by id → d
$id=E+id	*e$	Reduce by F → id
$id=E+F	*e$	Reduce by T → F
$id=E+T	*e$	Shift
$id=E+T*	e$	Shift
$id=E+T*e	$	Reduce by id → e
$id=E+T*id	$	Reduce by F → id
$id=E+T*F	$	Reduce by T → T*F
$id=E+T	$	Reduce by E → E+T
$id=E	$	A accept

c) a=b*c*d

Right Most Derivation

A → **id = E**
→ id = **T * F**
→ id = T * **id**
→ id = T * **d**
→ id = **T * F** * d
→ id = T * **id** * d
→ id = T * **c** * d
→ id = **F** * c * d
→ id = **id** * c * d
→ id = **b** * c * d
→ **a** = b * c * d

Handle Sequence:
a → b → id → F → c → id →F → T*F → d → id → T*F → id=E

Stack	**Input**	**Action**
$	a=b*c*d$	Shift
$a	=b*c*d$	Reduce by id → a
$id	=b*c*d$	Shift
$id=	b*c*d$	Shift
$id=b	*c*d$	Reduce by id → b
$id=id	*c*d$	Reduce by F → id
$id=F	*c*d$	Reduce by T → F
$id=T	*c*d$	Shift
$id=T*	c*d$	Shift
$id=T*c	*d$	Reduce by id → c
$id=T*id	*d$	Reduce by F → id
$id=T*F	*d$	Reduce by T → T*F
$id=T	*d$	Shift
$id=T*	d$	Shift
$id=T*d	$	Reduce by id → d
$id=T*id	$	Reduce by F → id
$id=T*F	$	Reduce by E → T*F
$id=E	$	A accept

d) a=a/b-c*d

Right Most Derivation

A → **id = E**
→ id = **E - T**
→ id = E – **T * F**
→ id = E – T * **id**
→ id = E – T * **d**
→ id = E – **F** * d
→ id = E – **id** * d
→ id = E – **c** * d
→ id = **T** – c * d
→ id = **T/F** – c * d
→ id = T/**id** – c * d
→ id = T/**b** – c * d
→ id = **F**/b – c * d
→ id = **id**/b – c * d
→ id = **a**/b – c * d
→ **a** = a/b – c * d

Handle sequence:
a → a → id → F → b → id → T/F → T → c → id → F → d → id → T*F → E-T → id=E

Stack	**Input**	**Action**
$	a=a/b-c*d$	Shift
$a	=a/b-c*d$	Reduce by id → a
$id	=a/b-c*d$	Shift
$id=	=a/b-c*d$	Shift
$id=a	/b-c*d$	Reduce by id → a
$id=id	/b-c*d$	Reduce by F → id
$id=F	/b-c*d$	Reduce by T → F
$id=T	/b-c*d$	Shift
$id=T/	b-c*d$	Shift
$id=T/b	-c*d$	Reduce by id → b
$id=T/id	-c*d$	Reduce by F → id
$id=T/F	-c*d$	Reduce by T → T/F
$id=T	-c*d$	Reduce by E → T
$id=E	-c*d$	Shift
$id=E-	c*d$	Shift
$id=E-c	*d$	Reduce by id → c

$id=E-id	*d$	Reduce by F → id
$id=E-F	*d$	Reduce by T → F
$id=E-T	*d$	Shift
$id=E-T*	d$	Shift
$id=E-T*d	$	Reduce by id → d
$id=E-T*id	$	Reduce by F → id
$id=E-T*F	$	Reduce by T → T*F
$id=E-T	$	Reduce by E → E-T
$id=E	$	A Accept

e) a=b+c-d-f

Right most derivation
A → **id = E**
→ id = **E – T**
→ id = E – **F**
→ id = E – **id**
→ id = E – **f**
→ id = **E – T** – f
→ id = E – **F** – f
→ id = E – **id** – f
→ id = E – **d** – f
→ id = **E + T** – d – f
→ id = E + **F** – d – f
→ id = E + **id** – d – f
→ id = E + **c** – d – f
→ id = **F** + c – d – f
→ id = **id** + c – d – f
→ id = **b** + c – d – f
→ **a** = b + c – d – f

Handle Sequence:
a → b → id → F → c → id → F → E+T → id → d → F → E-T → f → id → F → E-T → id=E

Stack	**Input**	**Action**
$	a=b+c-d-f$	Shift
$a	=b+c-d-f$	Reduce by id → a
$id	=b+c-d-f$	Shift
$id=	b+c-d-f$	Shift

$id=b	+c-d-f$	Reduce by id → b
$id=id	+c-d-f$	Reduce by F → id
$id=F	+c-d-f$	Reduce by E → F
$id=E	+c-d-f$	Shift
$id=E+	c-d-f$	Shift
$id=E+c	-d-f$	Reduce by id → c
$id=E+id	-d-f$	Reduce by F → id
$id=E+F	-d-f$	Reduce by T → F
$id=E+T	-d-f$	Reduce by E → E+T
$id=E	-d-f$	Shift
$id=E-	d-f$	Shift
$id=E-d	-f$	Reduce by id → d
$id=E-id	-f$	Reduce by F → id
$id=E-F	-f$	Reduce by T → F
$id=E-T	-f$	Reduce by E → E-T
$id=E	-f$	Shift
$id=E-	f$	Shift
$id=E-f	$	Reduce by id → f
$id=E-id	$	Reduce by F → id
$id=E-F	$	Reduce by T → F
$id=E-T	$	Reduce by E → E-T
$id=E	$	A Accept