Q) Three address code.

Quad.l

```
%{
  #include <stdio.h>
  #include "quad.tab.h"
  #include <string.h>
%}
%%
[a-z]([a-z]|[0-9])* {strcpy(yylval.exp, yytext);
  return VAR; }
[\t];
\n return 0;
.return yytext[0];
%%

int yywrap()
{
return 0;
}
```

Quad.y

```
%{
  # include <stdio.h>
  # include <string.h>
   struct quad
  {
   char op[5];
   char arg1[10];
   char arg2[10];
   char result[10];
  } QUAD[30];
```

```
int i=0, j;
%}
%union
{ char exp[10];
}
%token <exp> VAR
%type <exp> SETF
%%
S: E { printf("\n List of quadruples : ");
    for (j=0 ; j<i ; j++)
        printf("\n%s \t %s \t %s \t %s\n ";
            QUAD[j].op , QUAD[j].arg1,
            QUAD[j].arg2, QUAD[j].result);
    };

E: E'+'T {  strcpy (QUAD[i].op, "+");
            strcpy (QUAD[i].arg1, $1);
            strcpy (QUAD[i].arg2, $3);
            strcpy (QUAD[i].result, $$);

T{};

F: VAR{};
%%
main ()
{ yyparse();
  getchar();
}
yyerror (char *s) {
    printf("%s", s); }.
```

Output:-

a+b+c

List of quadruple:

*	+	b	a	t_1
	+	c	t_1	t_2