計算例

(1-2+3)/1*4//2
これを逆ポーランド記法にすると、１２- 3 + 1 / ４ * 2 //

1)
式から取り出したもの　　：1
残っている式　　：2 - 3 + 1 / ４ * 2 //
スタックの状態　　：[ ]
処理　　：スタックにpush

2)
式から取り出したもの　　：2
残っている式　　： - 3 + 1 / ４ * 2 //
スタックの状態　　：[1]
処理　　：スタックにpush

3)
式から取り出したもの　　：-
残っている式　　：3 + 1 / ４ * 2 //
スタックの状態　　：[1,2]
処理　　：スタックから2回popし，できた式をpush

3.1)
式の作成　　：(1+2)

4)
式から取り出したもの　　：3
残っている式　　：+ 1 / ４ * 2 //
スタックの状態　　：[(1+2)]
処理　　：スタックにpush

5)
式から取り出したもの　　：+
残っている式　　： 1 / ４ * 2 //
スタックの状態　　：[(1+2),3]
処理　　：スタックから2回popし，できた式をpush
5.1)
式の作成　　：((1+2)-3)

6)
式から取り出したもの　　：1
残っている式　　： / ４ * 2 //
スタックの状態　　：[((1+2)-3)]
処理　　：スタックにpush

7)
式から取り出したもの　　: /
残っている式　　:　４ * 2 //
スタックの状態　　: [((1+2)-3),1]
処理　　: スタックから2回popし，できた式をpush
また、/を読み込んだので、スタックから最後に読み込まれた数値に.to_fを付ける。

5.1)
式の作成　　: (((1+2)-3)/1to_f)

8)
式から取り出したもの　　: 4
残っている式　　:　* 2 //
スタックの状態　　: [(((1+2)-3)/1.to_f)]
処理　　: スタックにpush

9)
式から取り出したもの　　: *
残っている式　　:　 2 //
スタックの状態　　: [(((1+2)-3)/1.to_f),4]
処理　　: スタックから2回popし，できた式をpush
を付ける。

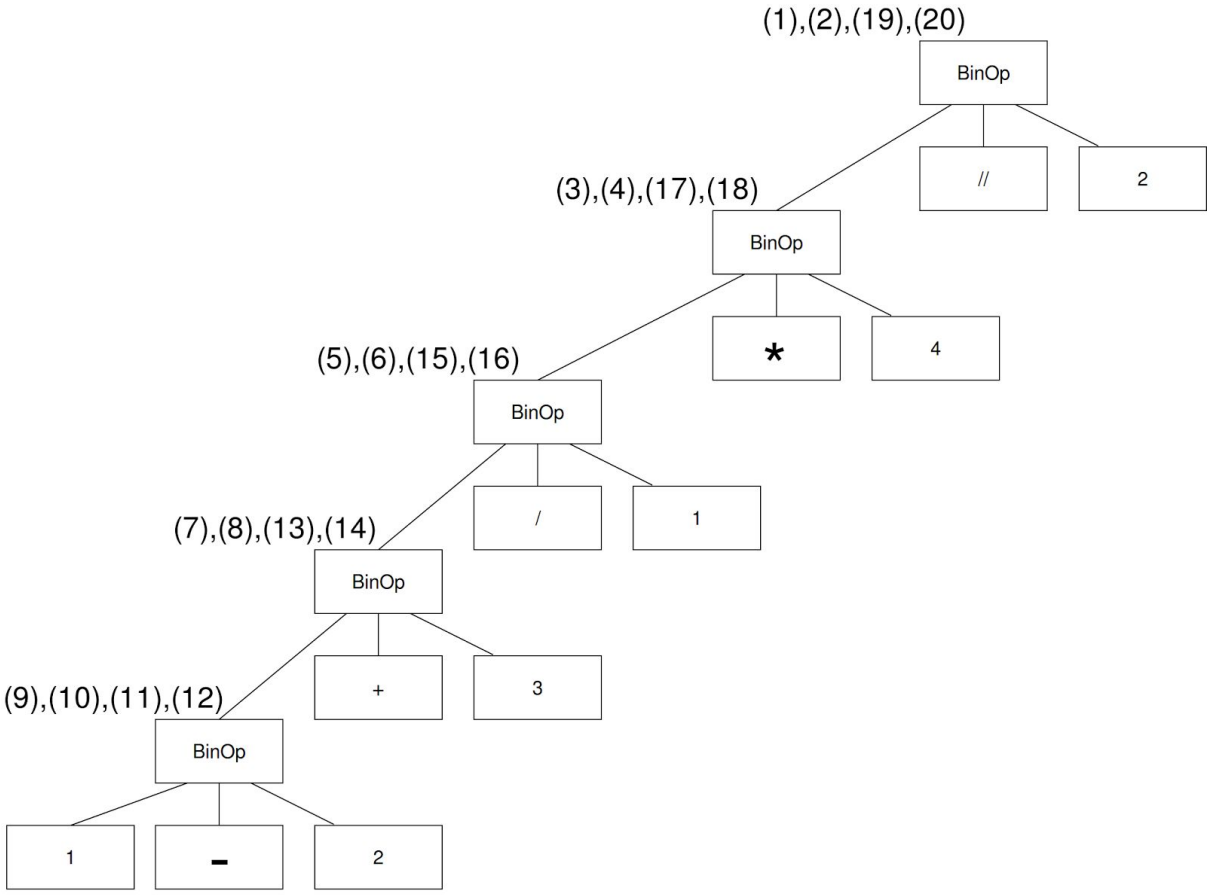
9.1)
式の作成　　: ((((1+2)-3)/1to_f)*4)

10)
式から取り出したもの　　: 2
残っている式　　:　 //
スタックの状態　　: [(((((1+2)-3)/1to_f)*4)]
処理　　: スタックにpush

11)
式から取り出したもの　　: //
残っている式　　:
スタックの状態　　: [(((((1+2)-3)/1.to_f)*4),2]
処理　　: スタックから2回popし，できた式をpush
を付ける。また、//を読み込んだので、/を出力する．次にスタックから最後に読み込まれた数値に.to_iを付ける。

11.1)
式の作成　　: (((((1+2)-3)/1to_f)*4)/2.to_i)

12)
式が残っていないので，スタックに残っている式が求める出力となる



(1)　'('を出力
　　出力：(
(2)　leftの中身を見て，BinOpなのでleftを引数として再帰
　　出力：(
(3) '('を出力
　　出力：((
(4) leftの中身を見て，BinOpなのでleftを引数として再帰
　　出力：((
(5) '('を出力
　出力：(((
(6) leftの中身を見て，BinOpなのでleftを引数として再帰
出力：(((
(7) '('を出力
　出力：((((
(8) leftの中身を見て，BinOpなのでleftを引数として再帰
出力：((((
(9) '('を出力
　出力：(((((

(10) leftの中身を見て、数値の1なので’1’を出力
出力：(((((1
(11) opの中身を見て、-なので-を出力
出力：(((((1-
(12) rightの中身を見て，数値の2なので’2’を出力．フラグは何もないので’)’を出力して終了
出力：(((((1-2)
(13) opの中身を見て、+なので+を出力
出力：(((((1-2)+
(14)rightの中身を見て，数値の3なので’3’を出力．フラグは何もないので’)’を出力して終了
出力：(((((1-2)+3)
(15) opの中身を見て、/なので/のフラグをセットして’/’を出力
出力：(((((1-2)+3)/
(16)rightの中身を見て，数値の1なので’1’を出力。’/’のフラグがあるので’1’の後に.to_fを出力して’)’を出力して終了
出力:(((((1-2)+3)/1.to_f)
(17)opの中身を見て、*なので*を出力
出力:(((((1-2)+3)/1.to_f)*
(18) rightの中身を見て，数値の4なので’4’を出力．フラグは何もないので’)’を出力して終了
出力:(((((1-2)+3)/1.to_f)*4)
(19) opの中身を見て、//なので//のフラグをセットして’/’を出力
出力:(((((1-2)+3)/1.to_f)*4)/
(20) rightの中身を見て，数値の2なので’2’を出力。’/’のフラグがあるので’2’の後に.to_iを出力して’)’を出力して終了
出力:(((((1-2)+3)/1.to_f)*4)/2.to_i)