```
#!/usr/bin/perl


use strict;
use warnings;
use Bio::SeqIO;
use Bio::Tools::SeqStats;

sub calc_freq_aa_strand{
    my($file,$aa)=@_;
    my $len_w=0;
    my $len_c=0;
    my $aa_w=0;
    my $aa_c=0;
    my $my_seqio=Bio::SeqIO->new( -file=>"$file", -format=>"fasta",-alphabet=>"protein");

    while(my $my_seq=$my_seqio->next_seq()){
        my $seq_stats=Bio::Tools::SeqStats->new(-seq=>$my_seq);
        my $hash_ref=$seq_stats->count_monomers();
        my $n_aa=$$hash_ref{$aa};
        my $id=$my_seq->display_id();
        my @explode=split(/ /,$id);
        my $gene=$explode[0];
        if($gene=~m/.{6}([W|C]).*/){
            if($1 eq "W"){
                my $len=$my_seq->length();
                $len_w+=$len;
                $aa_w+=$n_aa;
            }
            elsif($1 eq "C"){
                my $len=$my_seq->length();
                $len_c+=$len;
                $aa_c+=$n_aa;
            }
            else{print "$id skipped\n";}
        }
    }
    return($len_w,$aa_w,$len_c,$aa_c);
}

if ( $#ARGV != 0){
    print "No argument provided\n"; exit();
}

my $file=$ARGV[0];
my @aa=("T","S","Y");

foreach my $el (@aa){
    my($len_w,$aa_w,$len_c,$aa_c)=&calc_freq_aa_strand($file,$el);
    print "$el: $aa_w/$len_w; $aa_c/$len_c\n";
}
```