…This time though we’re going to scrape four, one for each of Florida’s four divisions.

```
base_4A <- "https://www.fhsaa.org/sites/default/files/orig_uploads/sports/swimming-diving/archives/2019-20/state/4A/191115F0"

base_3A <- "https://www.fhsaa.org/sites/default/files/orig_uploads/sports/swimming-diving/archives/2019-20/state/3A/191116F0"

base_2A <- "https://www.fhsaa.org/sites/default/files/orig_uploads/sports/swimming-diving/archives/2019-20/state/191108F0"

base_1A <- "https://www.fhsaa.org/sites/default/files/orig_uploads/sports/swimming-diving/archives/2019-20/state/1A/191109F0"

real_time_links <- function(base, event_numbers) { # function to make list of results links
  event_numbers <-
    sprintf("%02d", as.numeric(event_numbers)) # adds leading zeros where needed on event numbers
  links <-
    map(base, paste0, event_numbers, ".htm") # combines link base with event numbers
  links <- unlist(links, recursive = FALSE)
  return(links)
}

FL_Links <-
  real_time_links(base = c(base_4A, base_3A, base_2A, base_1A),
                  event_numbers = 1:24)

FL_Results <-
  map(FL_Links, read_results, node = "pre") %>% # map SwimmeR::read_results over the list of links
  map(swim_parse) %>%
  bind_rows() %>% # bind together results from each link
  select(Name, School, Finals_Time, Event) %>% # only the columns we need
  mutate(State = "FL") # add column for state since we'll be combining results with IL
```

Illinois Results

Illinois splits their results repositories into boys and girls. Final results from the Illinois girls state meet are just a scan of a printout, or as it’s sometime called .norm format. Not a good look Illinois, not a good look at all. Illinois will be penalized for this egregious data-foul by being made to use their preliminary results on the girls side. Frankly it’s the only way to include them in the meet at all, short of transcribing that miserable scan. Even I’m not that interested.

Both Illinois boys and Illinois girls (prelim) results are available as .pdf files, which `SwimmeR::read_results` can handle. There are some inconsistencies in spacing for class designators (“Sr”, “Jr” etc.) as well as a few leading spaces. There’s also an issue with a region name, which follos school names, but is placed in parentheses. We can clean these up with the `typo` and `replacement` arguments of `swim_parse`.

```
IL_Boys_Link <- "https://www.ihsa.org/data/swb/StateResults.pdf"
IL_Girls_Link <-
```

```
  "https://www.ihsa.org/Portals/0/prelim%20results.pdf"

IL_Results <-
  map(c(IL_Girls_Link, IL_Boys_Link), read_results) %>% # map
SwimmeR::read_results over the list of links
  map(
    swim_parse,
    avoid = c("IHSA", "NFHS", "POOL", "NATIONAL"),
    typo = c(
      "Sr\\s{2,}",
      # fix issue with some class designation strings
      "Jr\\s{2,}",
      "So\\s{2,}",
      "Fr\\s{2,}",
      "\\s{2,}\\(",
      # region designation fix
      "\\s\\d{1,2}\\s{2,}" # fix leading spaces
    ),
    replacement = c("Sr ",
                    "Jr ",
                    "So ",
                    "Fr ",
                    " \\)",
                    " ")
  ) %>%
  bind_rows() %>% # bind together results from each link
  mutate(Ind = case_when(str_detect(Event, "Relay") == FALSE ~ 1,
                         TRUE ~ 0)) %>%
  mutate(
    Grade = case_when(
      is.na(Grade) == TRUE &
        Ind == 1 ~ str_extract(School, "^Fr|^So|^Jr|^Sr")
    ),
    School = case_when(is.na(Grade) == FALSE ~ str_remove(School, Grade),
                       TRUE ~ School)
  ) %>%
  select(Name, School, Finals_Time, Event) %>% # only the columns we need
  mutate(State = "IL")
```

Joining Up Results

Having collected results from Florida and Illinois we just need to join them up, add a column for gender and remove events outside the standard 12 event program.

```
Results <- bind_rows(FL_Results, IL_Results) %>%
  mutate(Gender = case_when(
    str_detect(Event, "Girls") == TRUE ~ "Girls",
    # coding gender
    str_detect(Event, "Boys") == TRUE ~ "Boys"
  )) %>%
  filter(str_detect(Event, "Swim-off") == FALSE,
         # removing extra events
         str_detect(Event, "AWD") == FALSE)
```

Analysis

Last week to make a point about reusable code I copied my code from the week prior. Copy and pasting code is all well and good, but even better is to functionalize code. So this week we’re going to start working on function for scoring each event type (diving, relay and individual swim). The eventual goal, over the next few weeks, is to develop this code into a robust function for inclusion in a future release of the `SwimmeR` package.

Below is our starting point. It’s basically just the code from previous weeks wrapped in curly brackets. Inside the curly brackets there are still cases, determined by `if`, to break up the input `results` into diving, relays, and individual swims, and then to score them exactly as before. The difference is that this function does not assume there actually are any diving, relay or individual swims. It has `else` conditions following each `if` that insure the function does not fault if a particular event type isn’t represented.

Scoring Function

```
results_score <-
  function(results, events, point_values, max_place) {
    point_values <- c(point_values, 0)
    names(point_values) <- 1:length(point_values)

    results <- results %>%
      filter(str_detect(Event, paste(events, collapse = "|")) == TRUE)

    if (any(str_detect(results$Event, "Diving"))) {
      # break out diving results
      diving_results <- results %>%
        filter(str_detect(Event, "Diving")) %>%
        mutate(Finals_Time = as.numeric(Finals_Time)) %>%
        group_by(Event, Name) %>%
        slice(1) %>% # first instance of every diver
        ungroup() %>%
        group_by(Event) %>%
        mutate(
          Place = rank(desc(Finals_Time), ties.method = "min"),
          # highest score gets rank 1
          Finals_Time = as.character(Finals_Time)
        ) %>%
        filter(Place <= max_place)

    } else {
      diving_results  <-
        results[0, ] # if there are no diving results create a blank dataframe
called diving_results, with the same columns as results
    }

    if (any(str_detect(results$Event, "Relay"))) {
      # break out relay results
      relay_results <- results %>%
        filter(str_detect(Event, "Relay") == TRUE) %>% # only want relays
        group_by(Event, School) %>%
        slice(1) %>% # select first occurrence of team in each event
        ungroup() %>%
        mutate(Finals_Time_sec = sec_format(Finals_Time)) %>% # convert time to
seconds
        group_by(Event) %>%
        mutate(Place = rank(Finals_Time_sec, ties.method = "min")) %>% # places,
low number wins
        filter(Place <= max_place)
    } else {
      relay_results  <-
        results[0, ] # if there are no relay results create a blank dataframe
called relay_results, with the same columns as results
    }

    if (any(str_detect(results$Event, "Diving|Relay") == FALSE)) {
      # break out non diving/relay events
      ind_results <- results %>%
        filter(str_detect(Event, "Diving|Relay") == FALSE) %>%
        group_by(Event, Name) %>%
        slice(1) %>% # first instance of every swimmer
        ungroup() %>%
        group_by(Event) %>%
        mutate(Finals_Time_sec = sec_format(Finals_Time)) %>% # time as seconds
        mutate(Place = rank(Finals_Time_sec, ties.method = "min")) %>% # places,
low number wins
        filter(Place <= max_place)

    } else {
      ind_results <-
        results[0, ] # if there are no non relay/diving results create a blank
dataframe called ind_results, with the same columns as results
    }

    results <-
      bind_rows(ind_results, diving_results, relay_results) # bind together all
the dataframes.  This is why we needed blank dataframes in the case of no
relay/diving results

    results <- results %>% # deal with ties
      group_by(Event) %>%
      mutate(New_Place = rank(Place, ties.method = "first"),
             Points = point_values[New_Place]) %>%
      group_by(Place, Event) %>%
      summarize(Points = mean(Points)) %>%
      inner_join(results) %>%
      mutate(Points = case_when(str_detect(Event, "Relay") == TRUE ~ Points * 2,
                                TRUE ~ Points)) %>%
      ungroup()

  }
```

In future installments of the Swim-Off we'll add peices of code to this function to sanitize inputs, provide error messages, and eventually to score other meet formats.

Because `results_score` can score arbitrary events, we don't actually have to score an entire meet. We can select only a few events. Say we were interested in short axis strokes, butterfly and breaststroke. Those events can be passed to `results_score` in place of the full suite of events.

```
Results_Short_Axis <-
  results_score(
    results = Results,
    events = c("Butterfly", "Breaststroke"),
    point_values = c(20, 17, 16, 15, 14, 13, 12, 11, 9, 7, 6, 5, 4, 3, 2, 1),
    max_place = 16
  )

Results_Short_Axis %>%
  filter(Place <= 2) %>%
  select(Place, Name, State, Event, Points) %>%
  arrange(Event, Place) %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Place	Name	State	Event	Points
1	Iida, Max	IL	Boys 100 Yard Breaststroke	20
2	Alderson, Justin	IL	Boys 100 Yard Breaststroke	17
1	Filipovic, Aleksej	IL	Boys 100 Yard Butterfly	20
2	Jones, William	FL	Boys 100 Yard Butterfly	17
1	Gridley, Kaelyn	IL	Girls 100 Yard Breaststroke	20
2	Schwab, Carly	FL	Girls 100 Yard Breaststroke	17
1	Peoples, Olivia	FL	Girls 100 Yard Butterfly	20
2	Stone, McKenna	IL	Girls 100 Yard Butterfly	17

Scored Results

we are interested in scoring a full meet though, so let’s pass all the events from Florida vs. Illinois to `results_score` and get the scores for each event.

```
Results_Final <-
  results_score(
    results = Results,
    events = unique(Results$Event),
    point_values = c(20, 17, 16, 15, 14, 13, 12, 11, 9, 7, 6, 5, 4, 3, 2, 1),
    max_place = 16
  )

Scores <- Results_Final %>%
  group_by(State, Gender) %>%
  summarise(Score = sum(Points))
```

Florida wins both the boys and girls meets, but it’s close, especially on the boys side. On the girls side Illinois was likely hampered by not having their finals swims available. Serves them right for their lousy data practices.

```
Scores %>%
  arrange(Gender, desc(Score)) %>%
  ungroup() %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

State	Gender	Score
FL	Boys	1286
IL	Boys	1039
FL	Girls	1410
IL	Girls	915

```
Scores %>%
  group_by(State) %>%
  summarise(Score = sum(Score)) %>%
  arrange(desc(Score)) %>%
  ungroup() %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

State	Score
FL	2696
IL	1954

Let’s take a closer look, and see how many events each state won, by gender.

```
Results_Final %>%
  filter(Place == 1) %>%
  select(Event, State, Gender) %>%
  group_by(Gender, State) %>%
  summarise(Total = n()) %>%
  arrange(Gender, desc(Total)) %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Gender	State	Total
Boys	IL	11
Boys	FL	1
Girls	FL	7
Girls	IL	5

The Illinois boys won 11 of 12 events – that’s impressive! Let’s see that listed out:

```
Results_Final <-
  Results_Final[order(match(Results_Final$Event, Results$Event)),] # to order
events correctly

Results_Final %>%
  filter(Gender == "Boys",
         Place == 1) %>%
  select(Name, School, State, Finals_Time, Event) %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Name	School	State	Finals_Time	Event
	Oak Park (O.P.-River Forest)	IL	1:31.55	Boys 200 Yard Medley Relay
Maurer, Luke	Wilmette (Loyola Academy)	IL	1:36.35	Boys 200 Yard Freestyle
Iida, Max	Glenview (Glenbrook South)	IL	1:47.40	Boys 200 Yard IM
Boyle, Connor	Naperville (Neuqua Valley)	IL	20.04	Boys 50 Yard Freestyle
Williams, Jack	Flossmoor (Homewood-F.)	IL	544.2	Boys 1 mtr Diving

Name	School	State	Finals_Time	Event
Filipovic, Aleksej	St. Charles (North)	IL	48.25	Boys 100 Yard Butterfly
Boyle, Connor	Naperville (Neuqua Valley)	IL	43.82	Boys 100 Yard Freestyle
Andrew, Everet	Wilmette (Loyola Academy)	IL	4:24.50	Boys 500 Yard Freestyle
	LaGrange (Lyons)	IL	1:22.70	Boys 200 Yard Freestyle Relay
Zuchowski, Joshua	King's Academy	FL	47.85	Boys 100 Yard Backstroke
Iida, Max	Glenview (Glenbrook South)	IL	54.62	Boys 100 Yard Breaststroke
	Wilmette (Loyola Academy)	IL	3:02.80	Boys 400 Yard Freestyle Relay

Swimmers of the Meet

Just to remind everyone of the Swimmer of the Meet criteria first we'll look for athletes who won two events, thereby scoring a the maximum possible forty points. We'll also grab the All-American cuts to use as a tiebreaker, in case multiple athletes win two events.

```
Cuts_Link <-
  "https://raw.githubusercontent.com/gpilgrim2670/Pilgrim_Data/master/State_Cuts.csv"
Cuts <- read.csv(url(Cuts_Link))

'%!in%' <- function(x, y)
  ! ('%in%'(x, y)) # "not in" function

Cuts <- Cuts %>% # clean up Cuts
  filter(Stroke %!in% c("MR", "FR", "11 Dives")) %>%
  rename(Gender = Sex) %>%
  mutate(
    Event = case_when((Distance == 200 & #match events
                         Stroke == 'Free') ~ "200 Yard Freestyle",
                      (Distance == 200 &
                         Stroke == 'IM') ~ "200 Yard IM",
                      (Distance == 50 &
                         Stroke == 'Free') ~ "50 Yard Freestyle",
                      (Distance == 100 &
                         Stroke == 'Fly') ~ "100 Yard Butterfly",
                      (Distance == 100 &
                         Stroke == 'Free') ~ "100 Yard Freestyle",
                      (Distance == 500 &
                         Stroke == 'Free') ~ "500 Yard Freestyle",
                      (Distance == 100 &
                         Stroke == 'Back') ~ "100 Yard Backstroke",
                      (Distance == 100 &
                         Stroke == 'Breast') ~ "100 Yard Breaststroke",
                      TRUE ~ paste(Distance, "Yard", Stroke, sep = " ")
    ),

    Event = case_when(
      Gender == "M" ~ paste("Boys", Event, sep = " "),
      Gender == "F" ~ paste("Girls", Event, sep = " ")
    )
  )

Ind_Swimming_Results <- Results_Final %>%
  filter(str_detect(Event, "Diving|Relay") == FALSE) %>% # join
Ind_Swimming_Results and Cuts
```

```
  left_join(Cuts %>% filter((Gender == "M" &
                               Year == 2020) |
                              (Gender == "F" &
                                 Year == 2019)) %>%
              select(AAC_Cut, AA_Cut, Event),
            by = 'Event')

Swimmer_Of_Meet <- Ind_Swimming_Results %>%
  mutate(
    AA_Diff = (Finals_Time_sec - sec_format(AA_Cut)) / sec_format(AA_Cut),
    Name = str_to_title(Name)
  ) %>%
  group_by(Name) %>%
  filter(n() == 2) %>% # get swimmers that competed in two events
  summarise(
    Avg_Place = sum(Place) / 2,
    AA_Diff_Avg = round(mean(AA_Diff, na.rm = TRUE), 3),
    Gender = unique(Gender),
    State = unique(State)
  ) %>%
  arrange(Avg_Place, AA_Diff_Avg) %>%
  group_split(Gender) # split out a dataframe for boys (1) and girls (2)
```

Boys

As we've already seen, Illinois boys won 11 of 12 events, with Connor Boyle and Max Iida both winning two apiece. That means we'll be going to the All-American standard tiebreaker.

```
Swimmer_Of_Meet[[1]] %>%
  slice_head(n = 5) %>%
  select(-Gender) %>%
  ungroup() %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Name	Avg_Place	AA_Diff_Avg	State
Boyle, Connor	1.0	-0.025	IL
Iida, Max	1.0	-0.021	IL
Zuchowski, Joshua	1.5	-0.025	FL
Maurer, Luke	1.5	-0.022	IL
Tirheimer, Logan	3.0	-0.013	FL

It's Connor Boyle by a nose, just squeaking past Max Iida. Interestingly Florida's lone event winner, Joshua Zuchowski, had the same All-American score as Connor, but only won one event, finishing second to Max in the 200 IM. He's young though, so maybe next year.

```
Results_Final %>%
  filter(Name == "Boyle, Connor") %>%
  select(Place, Name, School, Finals_Time, Event) %>%
  arrange(desc(Event)) %>%
  ungroup() %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Place	Name	School	Finals_Time	Event
1	Boyle, Connor	Naperville (Neuqua Valley)	20.04	Boys 50 Yard Freestyle
1	Boyle, Connor	Naperville (Neuqua Valley)	43.82	Boys 100 Yard Freestyle

Girls

```
Swimmer_Of_Meet[[2]] %>%
  slice_head(n = 5) %>%
  select(-Gender) %>%
  ungroup() %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Name	Avg_Place	AA_Diff_Avg	State
Cronk, Micayla	1.0	-0.040	FL
Weyant, Emma	1.0	-0.033	FL
Cooper, Grace	1.5	-0.027	IL
Stege, Rachel	2.0	-0.020	IL
Novelline, Carly	2.5	-0.016	IL

The girls meet was overall much stronger, with the top three girls swimmers further under the All-American cuts than any of the boys. Micayla Cronk was especially excellent, not only winning two events, but doing so with extremely impressive times. Emma Weyant also won two events, but was pipped by Micayla on the All-American tie-breaker.

```
Results_Final %>%
  filter(Name == "Cronk, Micayla") %>%
  select(Place, Name, School, Finals_Time, Event) %>%
  arrange(desc(Event)) %>%
  ungroup() %>%
  flextable() %>%
  bold(part = "header") %>%
  bg(bg = "#D3D3D3", part = "header") %>%
  autofit()
```

Place	Name	School	Finals_Time	Event
1	Cronk, Micayla	Flagler	1:44.39	Girls 200 Yard Freestyle
1	Cronk, Micayla	Flagler	48.20	Girls 100 Yard Freestyle

In Closing

Thanks for joining us for another installment of the High School Swimming State-Off here…