SAS Base

- SAS[illegible] ([illegible]) [illegible]
- [illegible]
- [illegible] yyyymmdd_hhmmss [illegible]
- [illegible]
- [illegible]
- [illegible]
- proc sql [illegible]
- ods html [illegible]
- [illegible]
- data [illegible]
- [illegible]
- CASL
  - table.loadTable [illegible]
  - table.promote [illegible]

Base??????????

- マクロの世界と　　dataステップの世界を分けて考える
- シングルクォートとダブルクォートの違いをしっかり押さえる
- ステートメント、ステートメントに紐づくオプション、データセットに紐づくオプションをしっかり区別する
- dataステップが実際に動くときのなんとなくのイメージを掴む
    - 記述するのは列名とかだけであるが、実際の処理では　　1行ずつループされる
- 他のプログラミング言語とは異なり、「マクロを呼び出した箇所にマクロの定義内容を出力する」性質を持っている。そのため、カンマ区切りの内容だけを持つマクロを定義し、　　proc sqlの select句の指定部分に用いるなどの芸当が可能。

data???????(????????)

dataステップ内関数（文字列の情報取得）

count??

文字列Xに存在する'A'の個数をカウントする。 `count(X, 'A')`

dataステップ内関数（文字列の情報取得）

length??

文字列Xの長さを1バイト単位で取得する。 `length(X)`

dataステップ内関数 (文字列の情報取得)

klength??

文字列Xの長さを文字数単位で取得する（1バイト文字も2バイト文字も長さ1としてカウント）。

`klength(X)`

dataステップ内関数 (文字列の情報取得)

index??

文字列Xの最初の'ABC'の開始位置を取得する。 `index(X, 'ABC')`

dataステップ内関数(文字列の情報取得)

substr??

文字列Xの部分文字列（5～8文字目）を取得する。 `substr(X, 5, 8)`

data???????(??????)

data ステップ内関数　　(文字列の編集　　)

cat??

文字列を結合する。　　`cat(X, Y, ...)`

dataステップ内関数 (文字列の編集)

cats??

文字列の先頭・末尾のスペースを取り除き、結合する。

`cats(X, Y, ...)`

dataステップ内関数 (文字列の編集)

catx??

文字列の先頭・末尾のスペースを取り除き、指定した区切り文字で結合する。

`catx('区切り文字', X, Y, ...)`

dataステップ内関数（文字列の編集）

tranwrd??

- 文字列X内の'ABC'を'XYZ'に置換する。 `tranwrd(X, 'ABC', 'XYZ')`
    - `tranwrd(変換したい文字列, 検索文字列, 置換する文字列)`
    - TRANWRD関数 - SAS® Help Center

data ステップ内関数 (文字列の編集)

compress??

文字列 Xから、指定した文字 A
をすべて取り除く。除去対象の文字列を指定しない場合は、スペースをすべて取り除く。

```
compress(X, 'A')
```

data ステップ内関数　　(文字列の編集　　)

trim??

文字列の末尾のスペースを取り除く。　　`trim(X)`

strip??

文字列の先頭および末尾のスペースを取り除く。 `strip(X)`

data???????(???)

dataステップ内関数（その他）

fileexist??

- 外部ファイルの存在を確認する。存在する場合は`1`、存在しない場合は`0`が返却される。
  - FILEEXIST関数 - SAS® Help Center

??????????

グローバルステートメント

filename????????????

- 外部ファイル（　　.csvや .xlsx など）がある場所に「ファイル参照名」を付ける（パス情報を特別な変数に入れる）
  - filenameステートメント

```
filename test_data = 'C:\SAS_Study\test.csv';
proc import datafile=test_data out=work.test dbms=csv replace;
    getnames = yes;
    datarow = 2;
run;
```

□□□□□□□□□□□□

x?????????????

- OS□□□□□□□□□□
  - x□□□□□□□ - SAS® Help Center

libname????????????

- SASデータセットが格納される　　SASライブラリを定義する。
  - libnameステートメント
- データセット以外にも「　　SAS
  エンジン」を指定することで、他のタイプのファイルへのアクセスを可能にすることもできる。

proc????

PROC□□□□ □□□□ - SAS® Help Center

procステップ

proc sort???????

- データセットのソートを行う。
  - proc sortステートメント - SAS® Help Center

????

```
proc sort data=sashelp.cars out=work.cars_sorted;
by Model;
run;
```

?????

`proc sort`から　`;`までの間にスペース区切りで記述する。

オプション	指定する内容	説明
data=	データセット名	ソート対象のデータセットを指定
out=	データセット名	ソートした内容を格納するデータセットを指定（指定しない場合は上書きされる）
nodupkey	-	重複の削除を行う場合は指定（キー変数について、重複するデータがあった場合は重複している2件目以降の行を削除する）

?????????????

`proc sort ~;`から　`run;`までの行に記述する。

ステートメント	指定する内容	説明
by	列名	ソートのキーとする列を指定（列名の直前に`descending`を付与することで、降順でソートできる）

procステップ

proc transpose???????

- データセットを転置する。
  - proc transposeステートメント　　- SAS® Help Center

????

```
proc sort data=sashelp.cars out=work.cars_sorted;
by Model;
run;

proc transpose data=work.cars_sorted out=work.cars;
by Model;
run;
```

?????

`proc transpose`から　`;`までの間にスペース区切りで記述する。

オプション	指定する内容	説明
data=	データセット名	転置対象のデータセットを指定
out=	データセット名	転置後のデータセットを指定
prefix=	文字列	転置後の列名に付与したい接頭辞を指定
suffix=	文字列	転置後の列名に付与したい接尾辞を指定

?????????????

`proc transpose ~;`から　`run;`までの行に記述する。

ステートメント	指定する内容	説明
by	列名	転置せずにそのまま残す列を指定 （ by を使用する場合は、指定する列で事前に ソートしておく必要がある）
var	列名	転置する列を指定
id	列名	列の内容を「転置後の列名」として使用 する列を指定
idlabel	列名	転置後の変数ラベルとして使用する変数 を指定

procステップ

proc import???????

- 外部ファイルを読み込む。
  - proc importステートメント　　- SAS® Help Center

????

```
proc import datafile='C:\SAS_Study\test.csv' out=work.test dbms=csv replace;
    getnames = yes;
    datarow = 2;
run;
```

```
filename input_data = 'C:\SAS_Study\input_data.txt'
proc import datafile=input_data out=work.test dbms=tab replace;
    getnames = yes;
run;
```

?????

`proc import`から　`;`までの間にスペース区切りで記述する。

オプション	指定する内容	説明
datafile=	ファイルパス or ファイル参照名	「入力ファイルのフルパス」または「filename ステートメントで定義したファイル参照名」を指定
out=	データセット名	転置後のデータセットを指定
dbms=	データソース名	外部ファイルのデータの種類を指定 xlsx / csv / tab / dlm(delimiter ステートメントと併用　　)
replace	-	出力データセットを上書きする場合は指定

?????????????

`proc import ~;`から　`run;`までの行に記述する。

ステートメント	指定する内容	説明
getnames=	yes / no	列名情報をファイルから取得するかどうかを指定

ステートメント	指定する内容	説明
sheet=	Excelシート名	読み込むシート名を指定（dbms=xlsxの場合のみ有効）

procステップ

proc export???????

- データセットを外部ファイルへ出力する。
  - proc exportステートメント - SAS® Help Center

????

```
proc export data=sashelp.class outfile='C:\SAS_Study\class.csv' dbms=csv replace;
run;
```

```
proc export data=sashelp.class outfile='C:\SAS_Study\class.xlsx' dbms=xlsx replace;
    sheet = class;
run;
```

?????

`proc export`から　`;`までの間にスペース区切りで記述する。

オプション	指定する内容	説明
data=	データセット名	出力対象のデータセットを指定
outfile=	ファイルパス or ファイル参照名	「出力先ファイルのフルパス」または「filename ステートメントで定義したファイル参照名」を指定
dbms=	データソース名	外部ファイルのデータの種類を指定 xlsx / csv / tab / dlm(delimiter ステートメントと併用)
replace	-	出力データセットを上書きする場合は指定

?????????????

`proc export ~;`から　`run;`までの行に記述する。

ステートメント	指定する内容	説明
putnames=	yes / no	列名情報をファイルに書き出すかどうかを指定（デフォルトは yes）
sheet=	Excelシート名	読み込むシート名を指定（ dbms=xlsx の場合のみ有効）

procステップ

proc univariate???????

- 各種統計量を算出し、レポートを出力する。
  - univariateプロシジャ　　- SAS® Help Center

```
proc univariate data=sashelp.class normal plot;
    var height;
    output out=work.result n=n mean=mean std=std median=median;
run;
```

??????????

データセットオプション

??????????????

- dataステップの　　setステートメントなどで、データセット名の直後に　　`データセット名(~~~)`
  のように記述し、利用する。
  - 複数のオプションを指定する場合はスペースで区切る。
  - カテゴリ別のデータセットオプション　　- SAS® Help Center

```
data work.test;
set sashelp.class(keep=Name Height Weight where=(Height>60 and Weight>100));
run;
```

keep=???????????

保持したい列名をスペース区切りで記載する。ここに記載されていない列はそもそも読み込まれないため、
dataステップなどで使用不可となる。

drop=???????????

除去したい列名をスペース区切りで記載する。ここに記載した列はそもそも読み込まれないため、
ステップなどで使用不可となる。

data

データセットオプション

where=???????????

https://documentation.sas.com/doc/ja/pgmsascdc/v_061/ledsoptsref/p0ny9o8t8hc5zen1qn3ft9dhtsxx.htm

データセット行の取得条件を指定する。　条件部分は `()` で括り、 `where=(Height>60 and Weight>100)` のように記述する必要がある。（`keep=` や `drop=` の後に `where=` が処理される）

```
data work.test;
  set sashelp.class(where=(Height>60 and Weight>100));
run;
```

データセットオプション

rename=???????????

データセットの列名を異なる列名で読み込む。 `rename=(name1=new_name1 name2=new_name2)` のようにセットで指定する。 (`keep=`や `drop=`の後に `rename=`が処理される)

```
data work.test;
  set work.table1(rename=(name1=new_name1 name2=new_name2));`
run;
```

??????????

外部ファイルを読み込む

Excel??????????????????????

- `proc import`を使用する。

```
%macro ImportExcel(i_file_path=, i_sheet_name=, ods_output_name=);
    proc import datafile=&i_file_path out=&ods_output_name. dbms = xlsx replace;
        getnames = yes;
        sheet = &i_sheet_name.;
    run;
%mend ImportExcel;
```

外部ファイルを読み込む

??CSV?????proc import ??????????????????

```
proc import datafile='C:\SAS_Study\test.csv' out=work.work.test dbms=csv replace;
    getnames = yes;
    datarow = 2;
run;
```

```
filename input_data = 'C:\SAS_Study\input_data.txt'
proc import datafile=input_data out=work.test dbms= replace;
    getnames = yes;
run;
```

外部ファイルを読み込む

??CSV?????data???????????????????

```
filename sample "C:\SAS_Study\test2.csv";
data work.test;
    infile sample dim="," firstobs=2;
    input id:$8. name:$32. class:8.;
run;
```

外部ファイルを読み込む

???????????(proc import)

- proc importステートメント　　- SAS® Help Center

????

```
proc import datafile='C:\SAS_Study\test.csv' out=work.test dbms=csv replace;
    getnames = yes;
    datarow = 2;
run;
```

```
filename input_data = 'C:\SAS_Study\input_data.txt'
proc import datafile=input_data out=work.test dbms=tab replace;
    getnames = yes;
run;
```

?????

`proc import`から　`;`までの間にスペース区切りで記述する。

オプション	指定する内容	説明
datafile=	ファイルパス or ファイル参照名	「入力ファイルのフルパス」または「filename ステートメントで定義したファイル参照名」を指定
out=	データセット名	転置後のデータセットを指定
dbms=	データソース名	外部ファイルのデータの種類を指定 xlsx / csv / tab / dlm(delimiter ステートメントと併用　　)
replace	-	出力データセットを上書きする場合は指定

?????????????

`proc import ~;`から　`run;`までの行に記述する。

ステートメント	指定する内容	説明
getnames=	yes / no	列名情報をファイルから取得するかどうかを指定
sheet=	Excelシート名	読み込むシート名を指定（　　dbms=xlsx の場合のみ有効）

外部ファイルを読み込む

infile???????

- データセットとして読み込む外部ファイルの情報を指定する。
  - infileステートメント　　- SAS® Help Center

```
filename test_data "C:\SAS_Study\test2.csv";
data work.test;
    infile test_data dim="," firstobs=2;
    input id:$8. name:$32. class:8.;
run;
```

????????????

制御文字を文字リテラルで扱う

????CR

```
'0D'x
```

制御文字を文字リテラルで扱う

????LF

```
'0A'x
```

制御文字を文字リテラルで扱う

????

```
'09'x
```

proc sql??????????????????

`where`ステートメント内で`is missing`演算子を使用する。反転する場合は`is not missing`を使用する。

data??????????????????????

`missing(列名)`を使用する。反転する場合は`not missing(列名)`とする。

???????N??????????????

dataステップ内で `repeat(繰り返したい文字, N - 1);` を使用する。もしくは `N * '繰り返したい文字'` とする。

proc sql???????????

- 通常の　SQLのように記述できる。

data??????set??????????????????????

`set データセット名 CUROBS=一時変数名`で新しい列に行番号を挿入できる。

??????????????????

`proc sort`および　`nodupkey`を使用する。

```
/* carsデータセットのMake列の内容を、重複削除して取得する */
proc sort data=sashelp.cars(keep=Make) out=work.cars nodupkey;
    by logic_type;
run;
```

??????????????????

- dataステップ内では `put`、マクロ内では `%put`を使用する。
- データセット内の変数を出力する場合は、シングルクォートやダブルクォートで括る必要はなく、
  変数名をそのまま指定する。
- 複数の内容を連結して出力する場合、`put`は半角スペースで並べて記載、`%put`は `||`や `cats()`
  で明示的に結合して記載する。
- `put`
  でデータセットの列を並べて記載した際、不要な半角スペースが付与されてしまう場合は、列名の
  直後に `+(-1)`を記載しておく。（詳細は putステートメントのリファレンスに記載されている）

```
%let val = test;
%put &val.;
```

```
data _null_;
    set sashelp.class;
    put Name +(-1) ', ' Age ;
run;
```

??????????????????????n???????????

dataステップ内で `scan(文字列, n, 区切り文字(指定しない場合はあらゆる文字列で分割する))` を使用し、値を取り出す。

??????????????????data ??????????????????????

- 出力先データセット名に `_null_` を指定する。

```
data _null_;
  put 'ログ出力のテスト';
run;
```

????????????????

- `%do %while` などを組み合わせる。

```
%macro LoopTest();
    %let name_list = AAA BBB CCC DDD;
    %let index = 1;
    %let name = %scan(&name_list., &index., ' ');

    %do %while(not %sysevalf(%superq(name) =, boolean));
        %put &name.;

        %let index = %eval(&index. + 1);
        %let name = %scan(&name_list., &index., ' ');
    %end;
%mend LoopTest;
```

proc sql??????????????????????????????

- `into: マクロ変数名 separated by '区切り文字'`を用いる。
  - 複数列から出力する場合でも　　intoは 1つだけ。マクロ変数名にコロンを前置する
  - INTO句 - SAS® Help Center

```
proc sql noprint;
    select Name, Age
        into :name_list separated by ' ', age_list separated by ' '
        from sashelp.class
        where age > 12;
quit;
```

SAS????????????(????????)????????????????????

- `retain` や `end = eof` を活用する。
  - RETAINステートメント - SAS® Help Center

```
data _null_;
    set work.input_table end = eof;
    attrib
        input_file_list length = $1000.
        output_file_list length = $1000.
    ;
    retain input_file_list output_file_list;

    input_file_list = catx(' ', input_file_list, cats('input_', category, '.xlsx'));
    output_file_list = catx(' ', output_file_list, cats('output_', category, '.csv'));

    /* 最後まで行ったらマクロ変数に出力 */
    if (eof) then do;
        call symputx("input_file_list", input_file_list, 'G');
        call symputx("output_file_list", output_file_list, 'G');
    end;
run;
```

- `end=ー時変数名` 最後の行を読み込んだ時、ー時変数に「1」が入る
- `_N_` データステップの反復回数（処理中の行番号）が格納される
- `cats(値1, 値2, ...)` 引数値の両端の半角スペースを除いて結合する

??????????????

- `filename`および　`file`ステートメントを使用する。
  - `filename` ...「 SAS
    ファイル参照名」と「外部ファイルまたは出力デバイス」の関連付け・関連付けの解除を行う。
  - `file` ... putステートメントの出力先ファイルを指定する。

```
data _null_;
    filename OUTFILE "file_path";
    file OUTFILE encoding="utf-8"; /* ファイル内容を空っぽにする */
run;

data _null_;
    set sashelp.class;
    file OUTFILE MOD; /* ファイルに追記する */
    put Name;
run;
```

????yyyymmdd_hhmmss????

```
%local _timestamp;
data _null_;
    call symputx("_timestamp", put(today(), yymmddn8.) || '_' || compress(put(datetime(),
tod8.), ':'));
run;
```

??????????

- カテゴリ別のシステムオプション - SAS® Help Center

????????

```
option nosource nonotes;
```

- `nosource` ... ソースステートメント（実行している SASコードの内容）を SAS ログに表示しないようにする
- `nonotes` ... NOTE ログを表示しないようにする（エラーメッセージや警告メッセージは変わらず表示される）

????????

```
option symbolgen mprint mlogic;
```

- `symbolgen` ... マクロ変数の置換結果を表示する
- `mprint` ... マクロ実行時に行われる SASステートメントをログに出力する
- `mlogic` ... マクロ変数の評価式やマクロの実行開始、終了をログに出力する

????????????????

- `%do %while` などを組み合わせる。

```
%macro LoopTest();
    %let name_list = AAA BBB CCC DDD;
    %let index = 1;
    %let name = %scan(&name_list., &index., ' ');

    %do %while(not %sysevalf(%superq(name) =, boolean));
        %put &name.;

        %let index = %eval(&index. + 1);
        %let name = %scan(&name_list., &index., ' ');
    %end;
%mend LoopTest;
```

????????????????????????

- `length()`□□□ `klength()` □□□□□□□□□□□□□□□□□□□□□□□□□□□□□ □□
    - □□□□□□□□□□□□□□

```
data _null_;
    comment_text = strip(&comment_contents.);
    comment_len = (length(comment_text) - klength(comment_text)) / 2 + klength(comment_text);
    comment_line = '/*-' || repeat('-', comment_len - 1) || '-*/';
    put comment_line;
    put '/* ' "&comment_contents." ' */';
    put comment_line;
run;
```

proc sql??????????????????????????????????

- `into: □□□□□□ separated by '□□□□□'`□□□□□□□□□□□□□□□□□□□□ into□ 1 □□□□□□□□□□□□□□□□□□□□□□
  - INTO□ - SAS® Help Center

```
proc sql noprint;
    select Name, Age
        into :name_list separated by ' ', age_list separated by ' '
        from sashelp.class
        where age > 12;
quit;
```

ods html???????

- データセットやレポートを HTMLファイルへ出力する。
  - ods htmlステートメント

```
ods html file='C:\SAS_Study\class.html';
proc means data=sashelp.class;
run;
ods html close;
```

????????

- YYMMDDx出力形式____

```
data work.date_info;
now_datetime = datetime();
now_date = datepart(now_datetime);
now_time = timepart(now_datetime);

yyyymmddn = put(now_date, yymmddn8.); /* yyyymmdd(n:区切り文字なし) */
yyyymmddb = put(now_date, yymmddb8.); /* yyyymmdd(b:スペース区切り) */
yyyymmddc = put(now_date, yymmddc8.); /* yyyymmdd(c:コロン区切り) */
yyyymmddd = put(now_date, yymmddd8.); /* yyyymmdd(d:コロン区切り) */
yyyymmddp = put(now_date, yymmddp8.); /* yyyymmdd(p:ピリオド区切り) */
yyyymmdds = put(now_date, yymmdds8.); /* yyyymmdd(s:スラッシュ区切り) */
yyyymdd10 = put(now_date, yymmdd10.); /* yyyy-mm-dd */
ddmmmyyyy = put(now_date, date10.); /* ddmmmyyyy */
hhmmss = put(now_time, time.); /* hh:mm:ss */

year = year(now_date);
month = month(now_date);
day = day(now_date);

hour = hour(now_time);
min = minute(now_time);
second = second(now_time);

format until_time time8.;
until_day = input('20250101', yymmdd8.) - now_date;
until_sec = input('24:00:00', hhmmss8.) - now_time;
run;
```

data??????????

```
data work.test;
set sashelp.class;

height_cm = round(height * 0.0254, 0.01); /* inchをmに変換 */
height_str = put(height_cm, best8.); /* 数値を文字列に変換 */
height_str = tranwrd(height_str, '.', 'm'); /* 文字列を置換 */
height_str = compress(height_str); /* 空白を削除 */
name_height = cats(Name, '_', height_str, 'cm'); /* 文字列を結合 */
run;
```

??

- dataステップ内で byステートメントを適用すると、 byステートメントで指定した列ごとに「 first.列名」および「 last.列名」の列が生成される。
  - 使用したいデータセット列は、 `proc sort` などであらかじめソートしておく必要がある。
  - SASによる BYグループの処理方法
- それぞれ、下記の値が格納されている。
  - first.列名 ... byステートメントでグループ化された各行のうち、最初の行にのみ '1' が格納されており、それ以外は '0'
  - last.列名 ... byステートメントでグループ化された各行のうち、最後の行にのみ '1' が格納されており、それ以外は '0'
- この変数を使用すると、「最初だけ処理、最後だけ処理」といった処理を行わせることができる。
- `first.column_name` や `last.column_name` を使用し、 `if` 文で分岐させる。

```
proc sort data=sashelp.fish out=work.fish_sorted;
by species;
run;

data work.fish;
set work.fish_sorted(where=(not missing(Weight))); /* 欠損値の行を削除 */
by species; /* species列でグループ化 */

/* 計算用の列を初期化 */
if first.species then do;
count = 0;
weight_sum = 0;
end;

/* グループ内の行について、件数と合計を計算 */
retain count weight_sum;
count = count + 1;
weight_sum = weight_sum + Weight;

/* 平均値を計算し、行を出力 */
if last.species then do;
weight_avg = round(weight_sum / count, 0.1);
output;
end;
```

```
/* 出力したい列を指定 */
keep species count weight_sum weight_avg;
run;
```

CASL

CASL

table.loadTable?????

https://documentation.sas.com/doc/ja/pgmsascdc/v_061/caspg/cas-table-loadtable.htm

CASL

table.promote?????

https://documentation.sas.com/doc/ja/pgmsascdc/v_061/caspg/cas-table-promote.htm