Appendix 1:

```
setwd("e:/2015 spring/242/assignment/1/data/")

#find the start loction "==="
find_start =
      #
      #this function is to find the start of the file base on the apperance of
      #"===="
function(line){
  indexEql = grep('===', line)
  return(indexEql+1)
}



#length of each === combination
numEqul =
      #
      #caculate the length of all kind of "="s. Based on this value, we can manipulate the header
      #
function(line, start){
  num = nchar(unlist(strsplit(line[start-1], '\\s+')))
  num = num + 1
}

#get the name
getName =
  #
  #process header to get variable names
  #
function(line,start){
      num.end = cumsum(numEqul(line,start))
      num.start = c(1 , num.end[-length(num.end)]+1)
      substring(line[start-2],num.start,num.end)
  }

#find the number of equal sign length and names
checkEql =
      #
      #sometime there is no header, we can "borrow" one from files with the same year.
      #
function(file,line,start){
  if(length(start) == 0){
      start = grep("    1    ",line)[1]
      if(grepl("^man",file)){
```

```
            line.a = readLines(paste0("wo",file))
            num = numEqul(line.a,find_start(line.a))
            name = getName(line.a,find_start(line.a))
        }
        else{
            line.a = readLines(gsub("wo","",file))
            num = numEqul(line.a,find_start(line.a))
            name = getName(line.a,find_start(line.a))        }
    }

    else{
        num = numEqul(line, start)
        name = getName(line,start)
    }
    return(list(num,name,start))
}

#read the data into datafram
read.data =
        #
        #
function(file){
    line = readLines(file,encoding = "UTF-8")

    #men2009 this is for women/men2009 encoding problem
    line = gsub(intToUtf8(0xA0)," ",line)
    start = find_start(line)

    #women 2001 this is for women2001 no header problem
    num.name = checkEql(file,line,start)
    num = num.name[[1]]
    name = num.name[[2]]
    start = num.name[[3]]

    tt = textConnection(line[start:length(line)])
    data = read.fwf(tt, num, comment.char = "")
    close(tt)

    names(data) = name

    return(data)
}
```

```
#read data and save them as data frame
dir = list.files()

for(i in 1:12){
   assign(paste0("men",1998+i),read.data(dir[i]))
}
for(i in 13:24){
   assign(paste0("women",1998+i-12),read.data(dir[i]))
}

#From above, we can find there is a problem men8(women2006/men10Mile_2006). It's sticky "=",
so we will solve
#this problem (men8,women8)
stickCut =
      #
      #cut the sticky "===="
      #
function(filename){
   index = grep("Net", names(filename), ignore.case = TRUE)
   pro = names(filename)[index]
   cut.index = regexpr("Net", pro)[1]



   ss = sapply(1:nrow(filename),
          function(i)substring(filename[6][i,],c(1,cut.index),
c(cut.index1,nchar(as.character(filename[6][i,])))))

   Hometown = ss[seq(1,length(ss),2)]
   Nettime = ss[-seq(1,length(ss),2)]

   filename = cbind(filename,Hometown,Nettime)[-index]
}

men2006 = stickCut(men2006)
women2006 = stickCut(women2006)

#man2008 and women2008, we dont need "5 Mi" "10 km" and corresponding "pace" there
indexDlt = grep("5 Mi|10 km",names(women2008),ignore.case = TRUE)
women2008 = women2008[-c(indexDlt,indexDlt+1)]
men2008 = men2008[-c(indexDlt,indexDlt+1)]

#men2003 and women2003 still have problem    about location of "==="
#men2003
line = readLines(dir[5],encoding = "UTF-8")
```

```
start = find_start(line)
num = numEqul(line ,start)
num[7] = num[7]+1
name = getName(line,start)

tt = textConnection(line[start:length(line)])
men2003 = read.fwf(tt, num, comment.char = "")
close(tt)

names(men2003) = name

#women2003
line = readLines(dir[17],encoding = "UTF-8")
start = find_start(line)
num = numEqul(line ,start)
num[7] = num[7]+1
name = getName(line,start)

tt = textConnection(line[start:length(line)])
women2003 = read.fwf(tt, num, comment.char = "")
close(tt)

names(women2003) = name

#now put    all datframes into a list
datanames = objects()[grep("[man,women][0-9]{4}",objects())]
dataList = lapply(datanames, function(i) get(i))
names(dataList) = datanames


##Now we get all original dataframe. We need to next three thing:
#1. seperate Div/Tot into two parts
#2. extract the sign of "#"&"*"
#3. change time to total minutes
#4. make variables' names formal



#1
sepDIVTOT =
      #
      #this function aims at cutting sticky "DIV"&"TOT"
      #consider them as individual variables gains advantages in analysis.
      #
```

```
  function(dataframe){
    index = grep("div", names(dataframe), ignore.case = TRUE)

    if(length(index)!=0){
      #find the lines don't match the "**/**"pattern, delete them.
      dlt = 0
      for(i in 1:nrow(dataframe)){
        if(!grepl("\\d+/\\d+",as.character(dataframe[index][i,])))
          dlt = c(dlt,i)
      }
      dataframe = dataframe[-dlt,]

      divtot = sapply(1:nrow(dataframe),
                      function (i) strsplit(as.character(dataframe[index][i,]),'/') )

      Div = unlist(divtot)[seq(1,length(unlist(divtot)),2)]
      Tot = unlist(divtot)[-seq(1,length(unlist(divtot)),2)]
      dataframe = cbind(dataframe,Div,Tot)[-index]
    }
    else return(dataframe)
  }


#2. find #*, notice #* glued with time
findSign =
    #
    #find #*, the return value is character which indicate the sign.
    #
function(line){
    line = sapply(line, as.character)
    if(length(grep("#",line))!=0) return("#")
    else if(length(grep("\\*",line))!=0) return("*")
    else return(NA)
  }

bindSign =
    #
    #
function(file){
    SIGN = sapply(1:nrow(file), function(i) findSign(file[i,]))
    cbind(file, SIGN)
  }

#3. change time to total minutes
```

```
changeTim =
        #
        #change the time format to total minutes
        #
function(time){
        sum = 0
        time = gsub("\\#|\\*","",time)
        time = as.numeric(unlist(strsplit(time,":")))
        for(i in 1:length(time)){
           sum = sum + time[i]*60^(length(time)-i)
        }
        return(sum)
    }

procTime =
        #
        #find the names of variables which invovles time format value
        #change it into total minutes
        #
function(file){
        #haha = dataList[[8]]
        #dataframe = haha
        index = grep("Net|ti|gun|pace|km|mi",names(file),ignore.case = TRUE)
        time = list()
        for(j in 1:length(index)){
              time = sapply(1:nrow(file), function(i) changeTim(file[index[j]][i,]))
              file[index[j]] = time
        }
        return(file)
}


#4. make variables' names formal
proName =
        #
        #change all the letters to capital letters
        #regard gun time as time
        #
function(name){
    name = toupper(gsub(" ","",name))
    if(grepl("gun",name,ignore.case = TRUE)) name = "TIME"
    else if(grepl("net",name,ignore.case = TRUE)) name = "NET"
    else name = name
}
```

```
#use to loop to apply all above functions to our dataset list
for(i in 1:24){
   dataList[[i]] = sepDIVTOT(dataList[[i]]) #div tot

   dataList[[i]] = bindSign(dataList[[i]])    #sign

   dataList[[i]] = procTime(dataList[[i]]) #time


   name = names(dataList[[i]])              #name
   name = sapply(1:length(name), function(i) proName(name[i]))
   names(dataList[[i]]) = name

   #add YEAR variable
   YEAR = regmatches(names(dataList[i]),regexpr("\\d{4}",names(dataList[i])))
   YEAR = rep(YEAR,nrow(dataList[[i]]))
   dataList[[i]] = cbind(dataList[[i]],YEAR)

   #add GENDER variable
   GENDER = regmatches(names(dataList[i]),regexpr("[a-z]{3,5}",names(dataList[i])))
   GENDER = rep(GENDER,nrow(dataList[[i]]))
   dataList[[i]] = cbind(dataList[[i]],GENDER)
   print(i)
}

#6.change the class of variables : AG    PLACE NUM
changeClass =
      #
      #
function(fac){
      fac = as.numeric(as.character(fac))
   }

for(i in 1:24){
   dataList[[i]]$AG = changeClass(dataList[[i]]$AG)
   dataList[[i]]$PLACE = changeClass(dataList[[i]]$PLACE)
   if(!is.null(dataList[[i]]$NUM)) dataList[[i]]$NUM = changeClass(dataList[[i]]$NUM)
   else next
}

for(i in 1:24){
   if(!is.null(dataList[[i]]$TOT))
```

```
   {
      dataList[[i]]$TOT = gsub(" ","",dataList[[i]]$TOT)
      dataList[[i]]$DIV = gsub(" ","",dataList[[i]]$DIV)
   }
}

#
sapply(1:24, function(i) class(dataList[[i]]$DIV))

save(dataList, file = "data.RData")
```

Appendix 2:

```
library(ggplot2)
library(plyr)
library(stringr)
library(grid)
library(ggmap)
library(rgeos)
library(rgdal)
library(httr)
library(dplyr)
load("e:/2015 spring/242/assignment/1//data.RData")
load("e:/2015 spring/242/assignment/1//wholeData.RData")
#wholeData = do.call(rbind.fill,dataList)
wholeData$HOMETOWN = str_trim(wholeData$HOMETOWN)
wholeData$NAME = str_trim(wholeData$NAME)
#save(wholeData, file = "wholeData.RData")
temp = wholeData

####PART about age gender time and year
#GROUP THE AGE
#SEE HOW TO GROUP BASED ON VALUE OF VARIABLE "TOT"
ggplot(data=temp, mapping=aes(x=AG, y=TOT, colour = GENDER))+
   geom_point()+
   facet_wrap(~YEAR,scales = "free")+
   ggtitle("AGE GROUP")

#add a new variable age(category)
change =
function(age){
      if(is.na(age)) return("0")
      if(age<20) return("1")
      if(20<=age&age<40) return("2")
      if(40<=age&age<50) return("3")
      if(50<=age&age<60) return("4")
      if(60<=age&age<70) return("5")
      else return("6")
   }

#ADD NEW VARIABLES: age
age = sapply(1:nrow(temp), function(p) change(temp["AG"][p,]))
temp = cbind(temp,age)

#take a look at each variables
#age
```

```
#plot about age
ggplot(temp)+geom_histogram(aes(x=as.factor(AG)))+ggtitle("AGE HISTOGRAM")

#the number of people is increasing along the year.The proportion of each group of age remain the same
#except age2.
ggplot(temp)+geom_bar(aes(x=as.factor(YEAR), fill=age))+coord_polar()+ggtitle("AGE ALONG YEAR")

#the number of people is increasing along the year. Men increase not that much compared with women.
ggplot(temp)+geom_bar(aes(x=as.factor(YEAR),
fill=GENDER))+coord_polar()+ggtitle("GENDER ALONE YEAR")

#the number of people is increasin
ggplot(temp)+geom_bar(aes(x=factor(1), fill=as.factor(YEAR)))+coord_polar(theta = "y")+ggtitle("NUMBER OF RUNNER ALONG YEAR")


ggplot(data=temp, mapping=aes(x=TIME, y=PLACE, colour = as.factor(YEAR),shape=GENDER))+
  geom_point()+
  facet_wrap(~YEAR,scales = "free")+ggtitle("PLACE VS TIME")
#the plot shows NET TIME is used to decide the place since 2009
temp[which(temp$YEAR == 2009|temp$YEAR == 2010),]$TIME = temp[which(temp$YEAR == 2009|temp$YEAR == 2010),]$NET

#men run more fast than women.
#when we consider only the top100 runner in each year, the top1~3 player run faster each year for men.
#But this not happend when we consider about women.
ggplot(data=temp[which(temp$PLACE<=100),], mapping=aes(x=TIME, y=PLACE, colour = as.factor(YEAR),shape=GENDER))+
  geom_point()+
  facet_wrap(~YEAR,scales = "free")+ggtitle("PLACE VS TIME")



#plot about age and run time
ggplot(temp) + geom_density(aes(x = TIME, colour = age))+facet_wrap(~YEAR, scales = "free")+ggtitle("TIME VS AGE")
ggplot(temp) + geom_density(aes(x = TIME, fill = age))+facet_wrap(~YEAR,scales = "free")+ggtitle("TIME VS AGE")
```

```
####PART about foreigners
findForeign =
  function(hometown){
    if(grepl("( [A-Z]{2}$)|Usa|USA|US|Us",hometown)) return("no")
    if(is.na(hometown)) return("unknown")
    else return("yes")
  }
FOREIGN = sapply(1:nrow(wholeData), function(i) findForeign(wholeData[["HOMETOWN"]][i]))
temp = cbind(temp, FOREIGN)

#density plot about density of time

ggplot(temp)+geom_density(aes(x=TIME,col=FOREIGN)) +ggtitle("DENSITY POLT OF TIME(FOREIGNER)")

#increase of women larger than men
#however the change of total number of foreigner seems not correspond to total number of all runners.
ggplot(temp[which(temp$FOREIGN == "yes"&temp$YEAR != 2006),]) +
  geom_histogram(aes(x = as.factor(YEAR),fill = GENDER),position = "dodge")+ggtitle("YEAR(FOREIGNER)")

#creat dataset about foreigner
dataForgner = temp[which(temp$FOREIGN == "yes"&temp$YEAR!=2006),]
num.for = nrow(dataForgner)
summary(as.factor(dataForgner$HOMETOWN))

#There are total 441 foreigners and 155 among them are from kenya. And they are grades are perfect.
topPlace = temp[which(temp$PLACE ==1|temp$PLACE ==2|temp$PLACE ==3|temp$PLACE ==4),]
sort(table(topPlace$HOMETOWN),decreasing = TRUE)[1:5]

#Ken AND kenya are the same. AS WE ALL KNOW, people in these two areas are good at long-distance race.


####PART about state and city

#change city,state to longitude and latiitude
```

```
#CODE from stack overflow
#copy start
# httr's write_disk can act like a cache as it won't download if
# the file exists

GET("http://www.mapcruzin.com/fcc-wireless-shapefiles/cities-towns.zip",
      write_disk("cities.zip"))
unzip("cities.zip", exdir="cities")

# read in the shapefile
shp <- readOGR("cities/citiesx020.shp", "citiesx020")

# extract the city centroids with name and state

geo <-
  gCentroid(shp, byid=TRUE) %>%
  data.frame() %>%
  rename(lon=x, lat=y) %>%
  mutate(city=shp@data$NAME, state=shp@data$STATE)
##copy end

##Function to extract the names of city and state
statePro =
      #
      #
function(hometown){
  index = gregexpr("[A-Z]{2,2}",hometown)[[1]][1]
  if(!is.na(index)&index>0){
     hometown                =                str_trim(substring(hometown,c(1,index),c(index-
1,nchar(as.character(hometown)))))
  }
  else if(is.na(index)) return(NULL)
}

##
#find lat and log
findLoc =
  function(filename){
     location = sapply(1:nrow(filename), function(i) statePro(filename["HOMETOWN"][i,]))
     location = unlist(location)
     city = location[seq(1,length(location),2)]
     state = location[-seq(1,length(location),2)]
     return(c(city,state))
}
```

```
####
num = 2010 #change this num
xy.men =findLoc(dataList[[num-1998]])

plot(table(xy.men[(length(xy.men)/2):length(xy.men)]),xaxt ="n")
sta = table(xy.men[(length(xy.men)/2):length(xy.men)])
axis(1,at=1:length(sta),labels=names(sta),cex.axis = .35)

xy.men = sapply(1:(length(xy.men)/2),
                function(i) (geo %>% filter(city==xy.men[i],
state==xy.men[length(xy.men)/2+i]))[1:2])
xy.men = unlist(xy.men)

xy.women =findLoc(dataList[[num+12-1998]])
xy.women = sapply(1:(length(xy.women)/2),
                  function(i) (geo %>% filter(city==xy.women[i],
state==xy.women[length(xy.women)/2+i]))[1:2])
xy.women = unlist(xy.women)

library(rworldmap)
newmap <- getMap(resolution = "low")
plot(newmap, xlim = c(-130, -65), ylim = c(20,55 ), asp = 1)
points(xy.men[seq(1,length(xy.men),2)],xy.men[-seq(1,length(xy.men),2)],cex = .4,col = "blue")
points(xy.women[seq(1,length(xy.women),2)],xy.women[-seq(1,length(xy.women),2)],cex = .4,col
= "red")
######################3###
#change the num to get a set of plots, we can find something about changing of areas.

################################

##part about runner who took part in this competition several years
forName = temp
INDICT = forName$YEAR - forName$AG
forName = cbind(forName,INDICT)
forName = forName[order(forName$NAME,forName$INDICT, forName$HOMETOWN),]

test = forName[1:300,]
funName =
  function(data,k = 25 ,flag = 1,dif = 0,per = 0,time=0,sd_err = 0){
    if(k>nrow(data)) return(data.frame(per,dif,time,sd_err));
    name = data$NAME[k]
    if(grepl(name,data$NAME[k+flag],ignore.case = TRUE)){
      flag = flag+1
      funName(data,k,flag,dif,per,time,sd_err)
```

```
    }
    else {
      if(flag == 1){
        k = k+flag
        funName(data,k,flag,dif,per,time,sd_err)
      }
      else{
        dif = c(dif,data$TIME[k+flag-1]-data$TIME[k])
        per = c(per,data$NAME[k])
        time = c(time,flag)
        sd_err = c(sd_err,sd(data$TIME[k:(k+flag-1)]))
        k = k+flag
        flag =1
        funName(data,k,flag,dif,per,time,sd_err)
      }
    }
  }

options(expressions = 120000)
#because of error :stack overflow we just chose a small number of people to observe.
test = forName[103200:106400,]
oldMen =funName(forName)
oldMen[which(oldMen$time>10),]$per
temp[which(temp$NAME == "Adam Stolzberg"),]$TIME
temp[which(temp$NAME == "Ann Robb"),]$TIME
temp[which(temp$NAME == "Bruce Kirch"),]$TIME
temp[which(temp$NAME == "Charles Clark"),]$TIME
temp[which(temp$NAME == "Frank Jankoski"),]$TIME
temp[which(temp$NAME == "Mark Smith"),]$TIME
temp[which(temp$NAME == "Ronnie Wong"),]$TIME
temp[which(temp$NAME == "Sunny Fitzgerald"),]$TIME



##########################################3
index.S = which(wholeData$S == " !")
table(wholeData$AG[index.S])
plot(table(wholeData$AG[index.S]))
#from this, most seed players seem to be young ones.
```