*Prior to completing this assignment, download and read the* ***CPT168 Final Exam Project Instructions*** *located in Blackboard. Then download this Microsoft Word file and use it as a template to answer the following questions. Attach and upload your completed Progress Report file to the correct Blackboard assignment.*

1. **Student Name**: Bryan Bibb
2. **Day and Date** of this Progress Report: October 12, 2023
3. Write a **100 to 200 word narrative** discussing your understanding of the **CPT168 Final Exam Project** and how you can accomplish the expected results:

I understand this project to be an application used by the organizer of the event rather than the attendees. Therefore, it includes functions for adding, updating, and deleting attendees, as well as a variety of administrative reports. There are values in a configuration file that the organizer can update for their particular event. These values are used in the reports to generate lists of each type of attendee, a list of each type of meal, a financial report that adds up all costs and revenue to return a net value for money raised, and a seating chart.   
  
I began by creating a diagram on paper of the various functions, constants, and lists that are needed. Then under each function I noted the basic operation and how data would flow from the main function to the specific operations. After that, I turned those notes into pseudocode, pasted below. What is left is the actual coding, but I am hopeful that it should be manageable because this assignment is more of a logic problem than a matter of complex coding. The pseudocode (at least for now) seems to have a good logical flow.

1. Outline your **plan of action to complete** the **CPT168 Final Exam Project** on time:  
   These steps are cumulative, working from the simple to the complex. I will begin with a CSV file of values created to test the initial functions, before I write the one that adds to the CSV file.

* write the main function and initial\_display for the main menu and report menu
* write list\_names function and test by Oct 22
* write edit\_person function and test
* write delete\_person function and test by Oct 29
* write add\_person function and test
* write find\_person function and test by Nov 5
* write financial\_report function and test
* write seating\_chart function and test by Nov 12
* Comprehensive test process
* Complete by Nov 22 to avoid Thanksgiving holiday
* Submission by Nov. 25

Write a **hierarchy chart** to show how your projects functions will relate to each other. (Refer to Chapter 4 and other chapters in this course):  
  
A diagram of a person's work flow

Description automatically generated

1. Use **pseudocode** to show your plan for how your **CPT168 Final Exam Project** will process data (Refer to Chapter 3 and research online):

\*\*Main()\*\*

Initialize Global Constants

PEOPLE\_FILE = [people.csv]

SEATING\_FILE = [seating.csv]

PRICES\_FILE = [prices.csv]

MEAL\_PRICE = [22]

COST\_CHICKEN = []

COST\_BEEF = []

COST\_VEG = []

COST\_FISH = []

COST\_KOSHER = []

COST\_CHILD = []

COST\_PER\_SERVER = []

COST\_VENUE = []

continue = [y]

While true:

Call initial\_display

If the choice is 1:

update\_prices

If the choice is 2:

list\_names

If the choice is 3:

find\_person

If the choice is 4:

edit\_person

If the choice is 5:

delete\_person

If the choice is 6:

report\_display

If the choice is 0:

break

Print Goodbye

\*\*Initial\_Display\*\*

Print header

Print choices:

1. Update Prices

2. List names

3. Add attendee

4. Find attendee

5. Edit attendee

6. Remove attendee

7. Organizer Reports

0. Exit

Take user input = choice

Return choice

\*\*update\_prices\*\*

print header

load prices file to prices\_list

print list formatted

prompt for user input to update each of these

1. MEAL\_PRICE = [22]

2. CHILD\_DISCOUNT = [0.75]

3. COST\_CHICKEN = []

4. COST\_BEEF = []

5. COST\_VEG = []

6. COST\_FISH = []

7. COST\_KOSHER = []

8. COST\_CHILD = []

9. COST\_PER\_SERVER = []

10. COST\_VENUE = []

0. Cancel, exit to main menu

for each choice, update the value in the list

write updated list to prices file

\*\*list\_names\*\*

read people file to a list

display list with formatting

\*\*find\_person\*\*

read people file to a list

prompt for user input = searched\_person

if searched\_person is in list:

print record

print further choices - edit person or exit to main menu

prompt for user input

if edit:

edit\_person

else:

exit to main menu

else if not:

print error message, name not found

print further choices - try again or exit to main menu

if try again:

find\_person

if add person:

add\_person

if not:

exit to main menu

\*\*add\_person\*\*

load people to master list

prompt for input = entered\_name

if entered name is in the list:

print error - already in list

prompt for further input - try again or exit

if try again:

add\_person

else:

exit to main menu

else if name is not in the list

name = name

prompt for meal choice = meal

display meal options

if 1, meal = chicken

if 2, meal = beef

if 3, meal = vegetarian

if 4, meal = fish

if 5, meal = kosher

if 6, meal = soup and salad

if 7, meal = child

if 0 (cancel)

exit to main menu

display role options

if 1, role = member

if 2, role = guest

if 3, role = staff

if 4, role = VIP

if 5, role = musician

if 6, role = child

if 0, (cancel):

exit to main menu

prompt for extra donation = donation

if >0, add to donation total

write master list to csv

return to main

\*\*edit\_person\*\*

load people to master list

prompt for user input = name

find\_person(name)

if person is not in list:

print error - not in list

prompt for further input - try again or exit

if try again:

edit\_person

else:

exit to main menu

else prompt user input, edit = y/n

if y:

get index # of found name

prompt for new values

write new values to index #

write updated master list to csv

return to main

\*\*delete\_person\*\*

load people to master list

prompt for user input = name

find\_person(name)

if person is not in list:

print error - not in list

prompt for further input - try again or exit

if try again:

delete\_person

else:

exit to main menu

else prompt user input, delete = y/n

if y:

get index # of found name

delete found name

write updated master list to csv

return to main

\*\*report\_display\*\*

Print header

Print choices:

1. List all data

2. List members, guests, VIP and children

3. List staff and musicians

4. Meal report

5. Financial report

6. Seating chart

0. Cancel, back to main menu

Take user input = choice

read CSV to master list

If choice is 0:

back to main

Elif choice is 2:

extract all members, guests, VIP, and children

Elif choice is 3:

extract all staff and musicians

Elif choice is 4:

extract how many of each type of meal

Elif choice is 5:

financial\_report

Elif choice is 6:

seating\_chart

print list formatted

back to report\_display

\*\*financial\_report\*\*

Print header

load prices file

Calculate costs = cost\_list:

catering = cost of each meal \* number of each meal

servers needed:

if number of seated adults and children % 10 = 0:

servers = number / 10

else

servers= (number /10) + 1

server\_cost = servers \* COST\_PER\_SERVER

calculate cost\_sum =

catering + venue\_fee + server\_cost

Calculate revenue = revenue\_list:

paid\_adult\_meals \* MEAL\_PRICE

paid\_child\_meals \* (MEAL\_PRICE \* CHILD\_DISCOUNT)

donations

calculate revenue\_sum

total\_net = revenue\_sum - cost\_sum

print formatted cost\_list, revenue\_list, total\_net

\*\*seating\_chart\*\*

load CSV into master list

print header

Unpack (name and meal choice):

all children into child\_list

all staff and musicians into staff\_list

guests, members, VIPs

Slice lists

slice child\_list into even tables < 6

how many children?

how many tables?

create that many tables

divide list and put one part into each table

slice adult\_list into even tables < 8

how many adults?

how many tables?

create that many tables

divide list and put one part into each table

Print tables

Child tables 1 through

Adult tables 1 through

Servers and musicians area (one combined)