Toronto Metropolitan University

Faculty of Engineering & Architectural Science

Course Number	CPS 188
Course Title	Computer Programming Fundamentals
Semester/Year	Winter 2023
Instructor	Alex Ufkes
Section No.	17
Group No.	36
Submission Date	April 1, 2023
Due Date	April 2, 2023

Assignment Title	Term Project

Name	**Student ID**	**Signature***
Fareez Mir	501159372	FMir
Raymond Cao Jiang	501183087	Raymond C.J.
Sejuti Saha	501167884	
Sanjana Urba	501173408	Sanjana Urba

DIABETES IN CANADA

Unravelling The Stats & Facts
Using C Programming and GNU Plot

Introduction

Diabetes is a prevalent chronic disease in Canada, affecting millions of citizens each year. According to Statistics Canada, the number of diagnosed diabetes cases continues to rise, particularly among the aging population. Without distinction of gender, **over 3 million Canadians are currently affected**. To gain a deeper understanding of this critical health concern, C programming will be utilized to analyze data on the prevalence of diabetes among various age groups over multiple years, focusing on the four most populous provinces in Canada - *Alberta, British Columbia*, *Ontario* and *Quebec*. It will also determine which province has the highest and lowest percentage of diabetes and identify the provinces above or below the national average. In addition, GNU plot functionalities will be used to represent the data graphically. Using a systematic approach and drawing significant conclusions, this report contributes to a greater understanding of the disease's progression.

C Programming Calculations

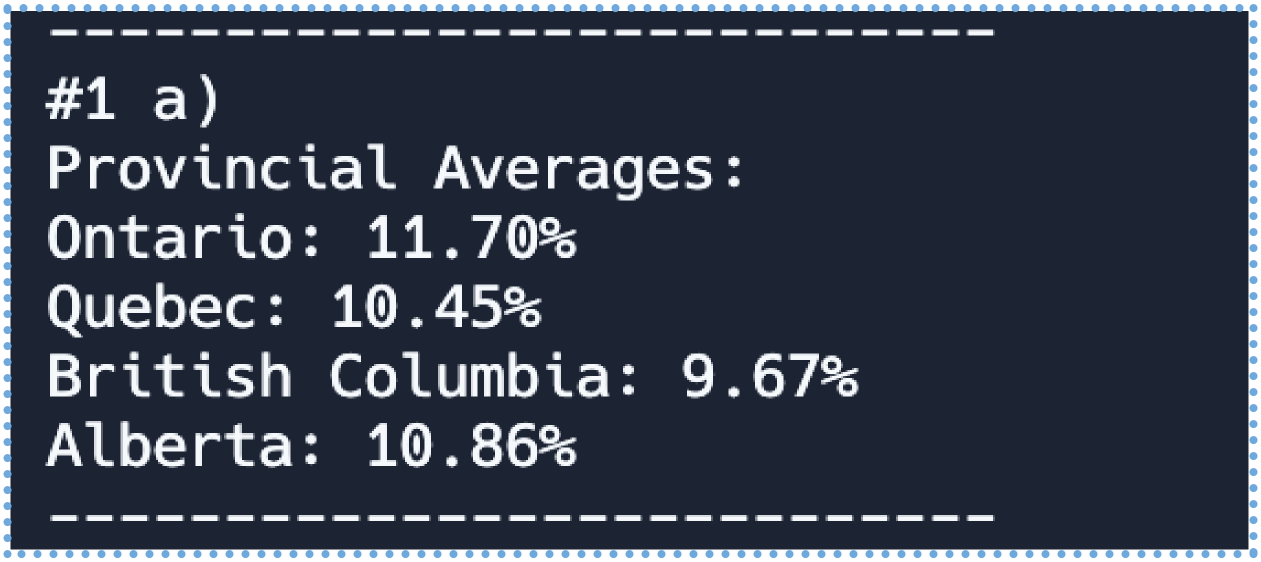
A thorough comprehension of how the C program works including the operations and functions used, is critical to understand the results produced.

Averages: Percentage of Population

Using data collected by Statistics Canada, the averages of the percentage of the population diagnosed with diabetes were represented on a geographical and annual scale, as well as by age group. In order for the program to accurately determine the various calculations, data was extracted from the data file into arrays. The array columns were defined as: **REF_DATE** (refers to years 2015-2021), **GEO** (refers to the province/country), **Age_group** (refers to age groups of 35-49, 50-64, and 65+), **Sex** (Female/Male), and **Value** (percentage of the population). The data file was opened and read (**"r"**) by the program and used **fgets** and **strtok** in a while loop to filter the data into their respective fields before they were copied using **strcpy** into their array line. Columns that were not relevant to the data needed were disregarded using a for loop. The copied values from **Value** were then converted from string values to double values using **atof** and stored into the **Value_Conv** array.

Provincial

The provincial diabetes percentage averages for *Alberta*, *British Columbia*, *Quebec*, and *Ontario* from 2015 to 2021 for groups aged 35 years and above were determined. The code begins with declaring and initializing the variables to add up each percentage (i.e. **ontario_total**) and count the number of percentages (i.e. **ontario_count**). A for loop is then used to iterate through the **GEO** array containing the province names and the **Value** array containing the diabetes percentage values. Within the loop, the **strcmp** function is used to compare the string stored in the **GEO** array with the given province. When the province is located in the array (indicated by the function returning a value of 0), the code uses the **strcmp** function again to check if the corresponding string stored in the **Value** array is not equal to "**F**", denoting no data entry. If not, the numerical value stored in the **Value_Conv** array is added to the total variable, while the count variable for the province is incremented by 1. This process is repeated for each province. By the end of the code, each province's total variable (i.e. **ontario_total**) contains the sum of all non-"**F**" percentages in the **Value_Conv** array, and its count variable (i.e. **ontario_count**) contains the number of percentages. The provincial average of diabetes is then found by dividing the total variable by the count variable and the 4 province results are printed and stored in the **p_*province*_avg** variables. *Figure 1* presents the output of the program.

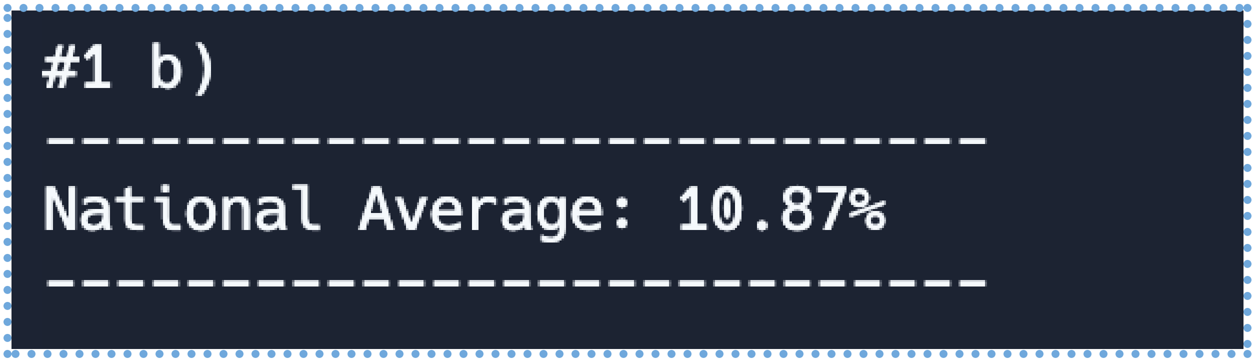


```
---------------------------
#1 a)
Provincial Averages:
Ontario: 11.70%
Quebec: 10.45%
British Columbia: 9.67%
Alberta: 10.86%
---------------------------
```

Figure #1 - Provincial Averages for each Province

National

The National (Canada excluding territories) average was found using the same method as above. A for loop was used to iterate through the **GEO** array containing the province names and **Value** arrays containing the diabetes percentage values. If statements are used along with the **strcmp** function to locate "*Canada (excluding territories)*" in the **GEO** array. If the corresponding string stored in the **Value** array is not equal to "**F**", denoting no data entry, the numerical value stored in the **Value_Conv** array is added to the **national_sum** variable, while the **national_count** variable is incremented by 1. The national average is determined by dividing the **national_sum** variable containing the sum of all numerical values in the **Value_Conv** array by the **national_count** variable. The **national_sum** is then divided by **national_count** to find the national average. The results were then printed as shown by *Figure 2*.

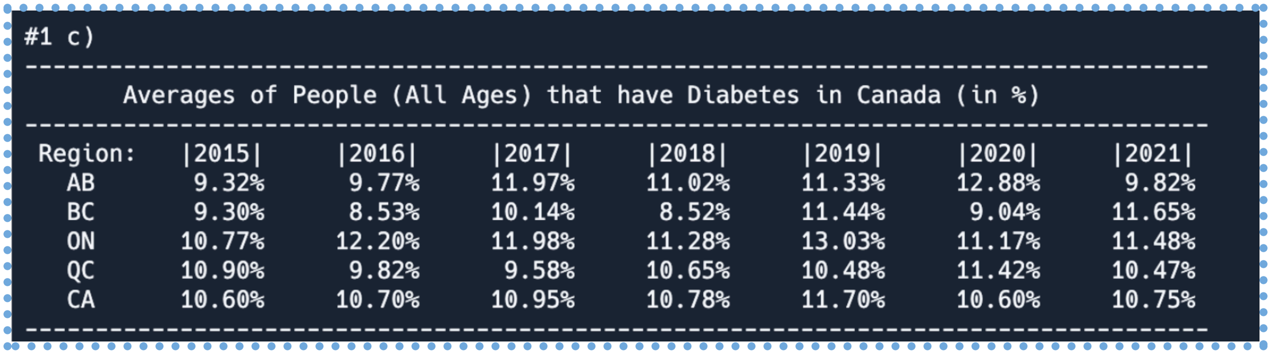


```
#1 b)
---------------------------
National Average: 10.87%
---------------------------
```

Figure #2 - National Average

Yearly Per Province

The yearly averages (2015-2021) were determined for each province and the whole country combining all age groups. The code begins by initializing arrays to store the regional yearly totals (i.e. **ontario_yearly_totals**) and regional yearly counts (i.e. **ontario_yearly_counts**) for each province and *Canada (excluding territories).* Although not required for this section, **yearly_totals** and **yearly_count** were also initialized to be used for question 4. A for loop is then used to iterate through the 7 years. Within the loop, the years are converted to a string and an index variable is initiated which takes the difference of 2015 from the current year (i.e. current - 2015). Since the loop starts at 2015 and the **yearly_totals** and regional arrays have 7 entries (one for each year from 2015 to 2021), this calculation will give an index value between 0 and 6, which corresponds to the appropriate entry in these arrays for the current year. By using the **index** variable to access the correct position in the arrays, the loop can correctly store and update the yearly totals and counts for each region and the country for each year. If statements are then used for each region, along with the **strcmp** function, to locate the current year in the **REF_DATE** array and locate the region in the **GEO** array. The statement also checks that the **Value** array does not contain an "**F**" indicating no percentage entry. The numerical values in the **Value_Conv** array are then added and stored in the specified yearly totals index array for each region (i.e. **ontario_yearly_totals[index]** += **Value_Conv[i]**). The regional yearly count variable is also incremented by 1 to count the number of percentages (i.e **ontario_yearly_counts[index]**++). Outside of the if statements, **yearly_totals** and **yearly_count** were also tallied and stored using the same process as before. This process is repeated for each year (since the index value changes) and for each province. The averages were found by dividing the regional yearly totals values in the regional yearly total arrays (i.e **ontario_yearly_totals[i]** by the regional count arrays (i.e. **ontario_yearly_counts[i]**) and were then printed as shown by *Figure 3*.

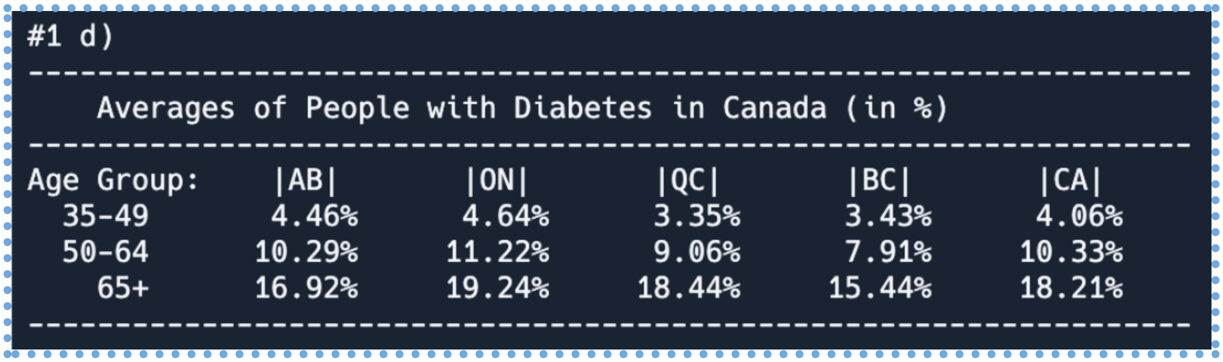


```
#1 c)
---------------------------------------------------------------------------------
       Averages of People (All Ages) that have Diabetes in Canada (in %)
---------------------------------------------------------------------------------
 Region:   |2015|     |2016|     |2017|     |2018|     |2019|     |2020|     |2021|
   AB       9.32%      9.77%     11.97%     11.02%     11.33%     12.88%      9.82%
   BC       9.30%      8.53%     10.14%      8.52%     11.44%      9.04%     11.65%
   ON      10.77%     12.20%     11.98%     11.28%     13.03%     11.17%     11.48%
   QC      10.90%      9.82%      9.58%     10.65%     10.48%     11.42%     10.47%
   CA      10.60%     10.70%     10.95%     10.78%     11.70%     10.60%     10.75%
---------------------------------------------------------------------------------
```

Figure #3 - Yearly Averages for each Region

Age Groups

The average percentage of diabetes among three age groups (*35-49, 50-64, 65+*) for all years and for each province were calculated. The 2D arrays, **yearly_age_totals** and **yearly_age_counts,** use rows to represent the provinces (5 total including National, going in order of: *Alberta, Ontario, Quebec, BC, Canada*), while the number of columns represent the 3 age groups. The code then enters a nested for loop, where the outer loop iterates through years from 2015 to 2021 and the inner loop iterates through the percentage values. If statements and **strcmp** is used to locate the age groups within the **Age_group** array and assign a value to each going up by 1 (i.e. 35-49 years has a value of 0, 50-64 has a value of 1, etc.). The age groups are split up such that in each section, the **strcmp** is used to locate the province in the **GEO** array and adds the numerical percentage value (ensuring that "**F**" or invalid data entry values are not included) to the corresponding array for the age group and province. For example, for the 35-49 age group in Alberta for all years, the percentage and count values would be added to the [0][0] position of their respective arrays. The count array values (i.e. **yearly_age_counts**) are then increased by 1. This process repeats for each age group for each province. Each average of the regions is found by dividing the **yearly_age_totals** over the **yearly_age_counts**, and are iterated using a for loop to extract the data for each of the age groups for their regions. The results are printed such that each of the averages are displayed corresponding to their region along with their age group. The subsequent results were printed as shown by *Figure 4*.



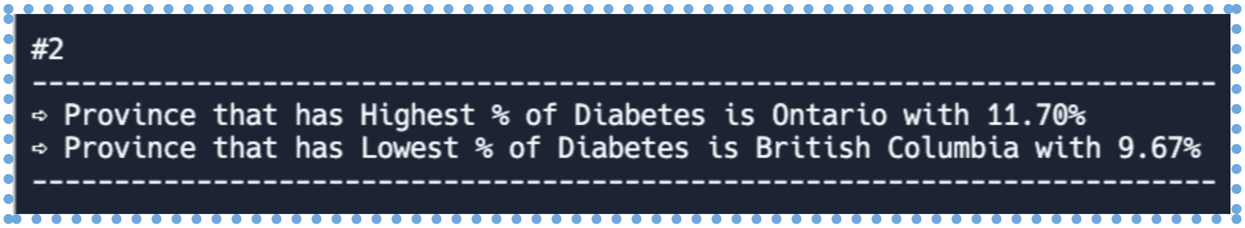
```
#1 d)
------------------------------------------------------------------
    Averages of People with Diabetes in Canada (in %)
------------------------------------------------------------------
Age Group:    |AB|       |ON|       |QC|       |BC|       |CA|
  35-49       4.46%      4.64%      3.35%      3.43%      4.06%
  50-64      10.29%     11.22%      9.06%      7.91%     10.33%
    65+      16.92%     19.24%     18.44%     15.44%     18.21%
------------------------------------------------------------------
```

Figure #4 - Age Group Averages for respective Regions

Comparisons & Analysis

Provinces with the Highest and Lowest Percentage

In order to determine the province with the highest percentage of diabetes, an array containing the averages of each province (amongst all age groups and all years) titled **province_avg** was initialized to hold 4 values, one for each province average (i.e. **province_avg[0] = p_ontario_avg)**. Another array was initialized containing the pointer of the 5 provinces and the national average named **province_names**. Each province in the **province_names** array is a string that is represented as a pointer to the first character in the sequence. Then a for loop is used to iterate through the **province_avgs** array from index 1 to the last index (i.e. **province_avgs_size - 1**). For each index **i**, if the percentage at index **i** is less than the percentage at index **min**, the **min** variable is updated to **i**. Subsequently, if the percentage at index **i** is greater than the percentage at index **max**, the **max** variable is updated to **i**. Finally, the province with the highest percentage of diabetes is printed by looking up the name in the **province_names** array using the max index and the corresponding percentage from the **province_avgs** array. This same procedure is used to find the province with the lowest percentage of diabetes, and are both shown in *Figure 5*.

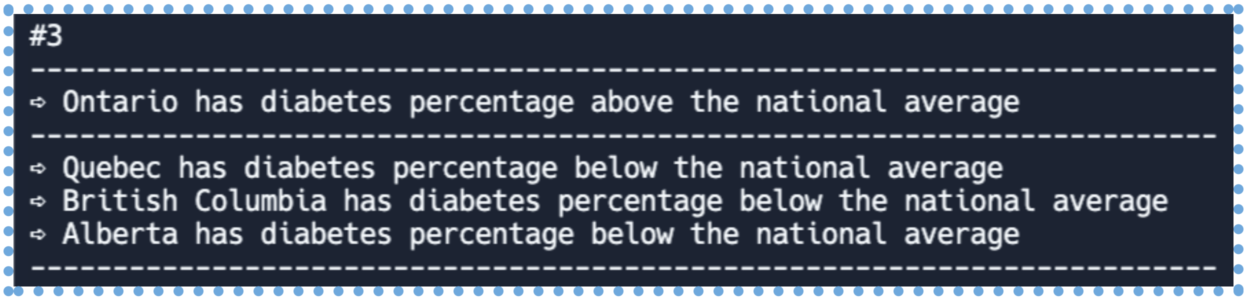


```
#2
------------------------------------------------------------------------
⇨ Province that has Highest % of Diabetes is Ontario with 11.70%
⇨ Province that has Lowest % of Diabetes is British Columbia with 9.67%
------------------------------------------------------------------------
```

Figure #5 - Highest and Lowest Provincial Averages

Provinces Above and Below the National Average

To find the provinces that were below above or below the national average, a for loop was used to iterate through each province in the **province_avgs** array. For each province, the code compares its diabetes percentage to the national average (i.e. **p_national_avg**). The code then prints a message stating the province's name and that the percentage is greater than the national average, or prints a message stating the province’s name and percentage is below the national average. The output of the program is illustrated by *Figure 6*.

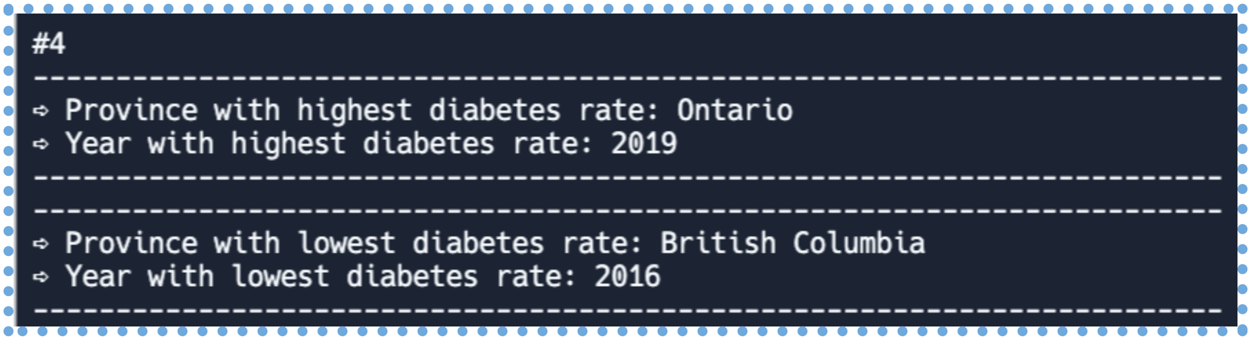


```
#3
------------------------------------------------------------------------
⇨ Ontario has diabetes percentage above the national average
------------------------------------------------------------------------
⇨ Quebec has diabetes percentage below the national average
⇨ British Columbia has diabetes percentage below the national average
⇨ Alberta has diabetes percentage below the national average
------------------------------------------------------------------------
```

Figure #6 - Provinces Above and Below the National Average

Year and Province with the Highest and Lowest Percentage

The year and the province with the highest and lowest diabetes percentage was found. This was done by using a for loop to iterate over an array of yearly totals and counts for the 7 years. For each year, the yearly average is calculated and it is compared to the current maximum yearly average. If statements are used to determine if the current yearly average is higher than the current maximum yearly average, in which case, the maximum yearly average is updated as well as its corresponding year. The province with the highest diabetes rate is determined by iterating over arrays of yearly totals and counts for 7 years for four provinces (*Alberta*, *Ontario*, *Quebec*, and *British Columbia*). Within the loop, the code calculates the average for each province by adding the yearly total divided by the yearly count. For example, for *Alberta*, the code adds the value **alberta_yearly_totals[i] / alberta_yearly_counts[i]** to the **alberta_avg** variable. After the loop, the code divides each province's total by 7, which is the number of years being averaged, to obtain the average for each province (e.g. **alberta_avg** /= 7). It then compares the current province's average to the current maximum province's average using pointer arithmetic. If the current province's average is higher than the current maximum province's average, it updates the **max_province_avg** and the corresponding province (e.g. **max_province** = **"Alberta"**). The year and province with the lowest percentage of diabetes is found using the same method, except using the **min** variables. The results are then printed, as exhibited by *Figure 7.*



```
#4
----------------------------------------------------------------------
➪ Province with highest diabetes rate: Ontario
➪ Year with highest diabetes rate: 2019
----------------------------------------------------------------------
----------------------------------------------------------------------
➪ Province with lowest diabetes rate: British Columbia
➪ Year with lowest diabetes rate: 2016
----------------------------------------------------------------------
```

Figure #7 - Province and Year with Highest and Lowest Percentage

Graphical Representations

Graphical iterations of the data allows for further interpolations to be made about the effects of diabetes on the population. GNUPlot was used to create two depictions of the data; a linear plot and a bar graph. The graphs summarize and compare the percentages geographically over time as well as in terms of age group.

GNU Plot Analysis

Linear Plot

Diabetes percentages for the accumulated age and gender groups for the four provinces along with the national average were represented in variation with time. The graph was first formatted through various **set** functions in order to customize elements such as the graph title, axis titles, x and y ranges, labels, and ticks. The data was input in a CSV file format, containing the years (2015-2021) along with their respective provincial averages and the national average in specified columns. Under the **plot** function, **u** was used to read the percentage data by column from the file with respect to the year, in terms of 'x' and 'y'. For example, to plot the "*Ontario*" line, **u 1:2** refers to column 1 as the x-value (year) and column 2 as the y-values (percentages). The lines and plot circles were customized with **w lp** (plot circle), **lw** (line width), **lt** (line type), and **lc** (colour). Each line was titled in the legend using **title**. The resulting graph is displayed by *Figure 8*.

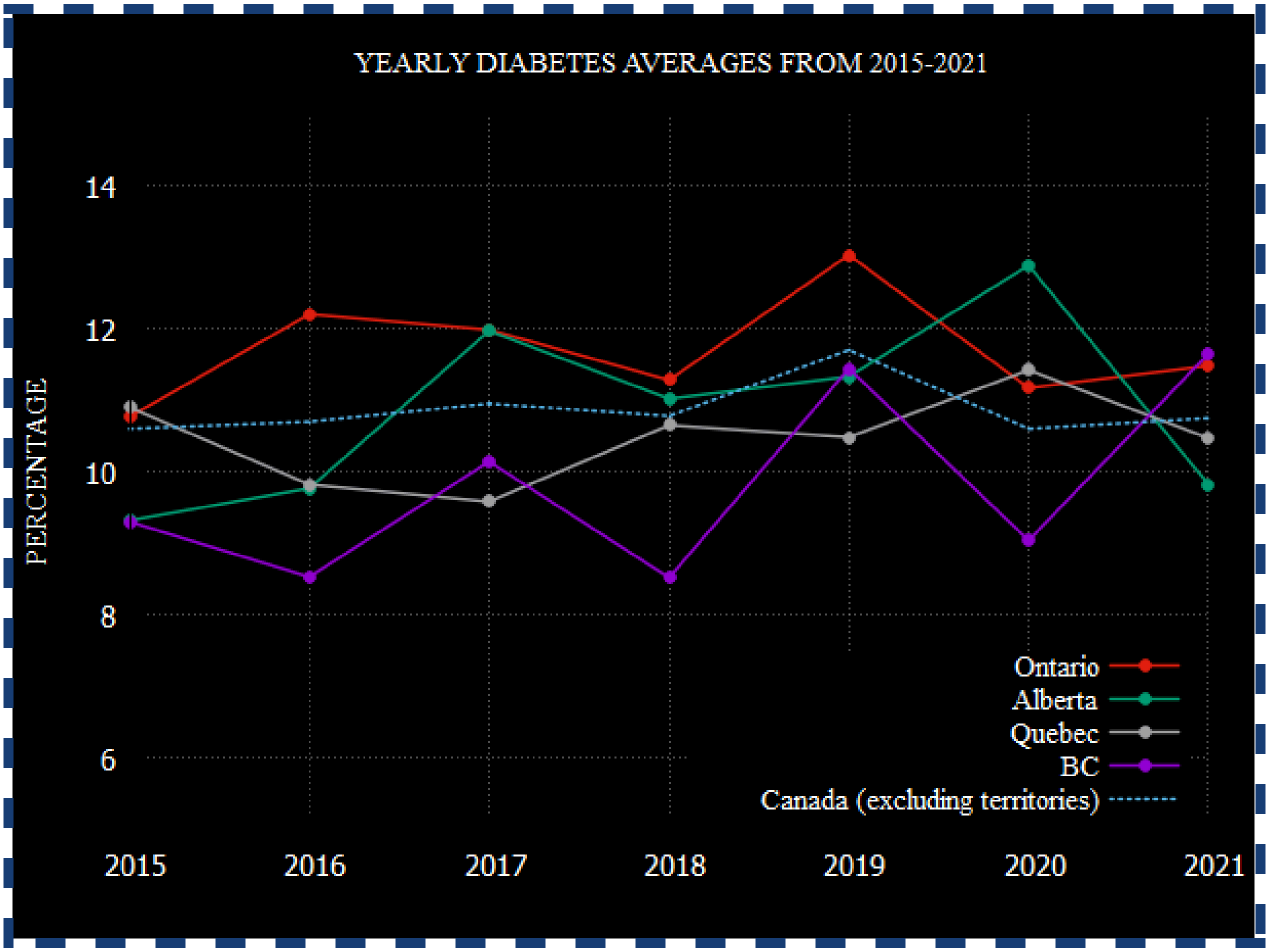


Figure #8 - Provincial Percentage Averages from 2015-2021 Presented in a Linear Plot

From the constructed graph, it can be seen that the percentages for all the provinces, although fluctuating, remain somewhat consistent throughout the years. Overall increases from 2015 to 2021 are minimal, ranging from 0.5 to 2.35 percent across provinces. The national average comparably reflects this trend, depicting an overall increase of 0.15%.

Bar Graph

The percentages were illustrated in terms of the three age groups for the entirety of Canada (excluding territories). Again, the graph was formatted through various **set** functions, this time accounting for the width of the bars with **boxwidth**. The CSV file was formatted so that the three age groups in each row were numbered (1-3) in column 1. The percentage for each age group “stood alone” in their column with separated commas (delimiters). The last column labeled the age group. Under the **plot** function, each bar was graphed with the **using** function where the first and second parameter refers to column 1 and 2 as the ‘x’ and ‘y’ values. The third parameter being **xtic(5)**, ensures that each series will be labeled on the x-axis with the data from column 5 (age group). For example, to construct the “35-49” age group bar, the function **using 1:2:xtic(5)**, refers to column 1 for its numbered age group, column 2 for its associated percentage value, and column 5 to label the series with “35-49”. and the The series were then titled in the legend with **title**, and represented as bar graphs through the **with boxes** function. The resulting graph is shown by *Figure 9*.

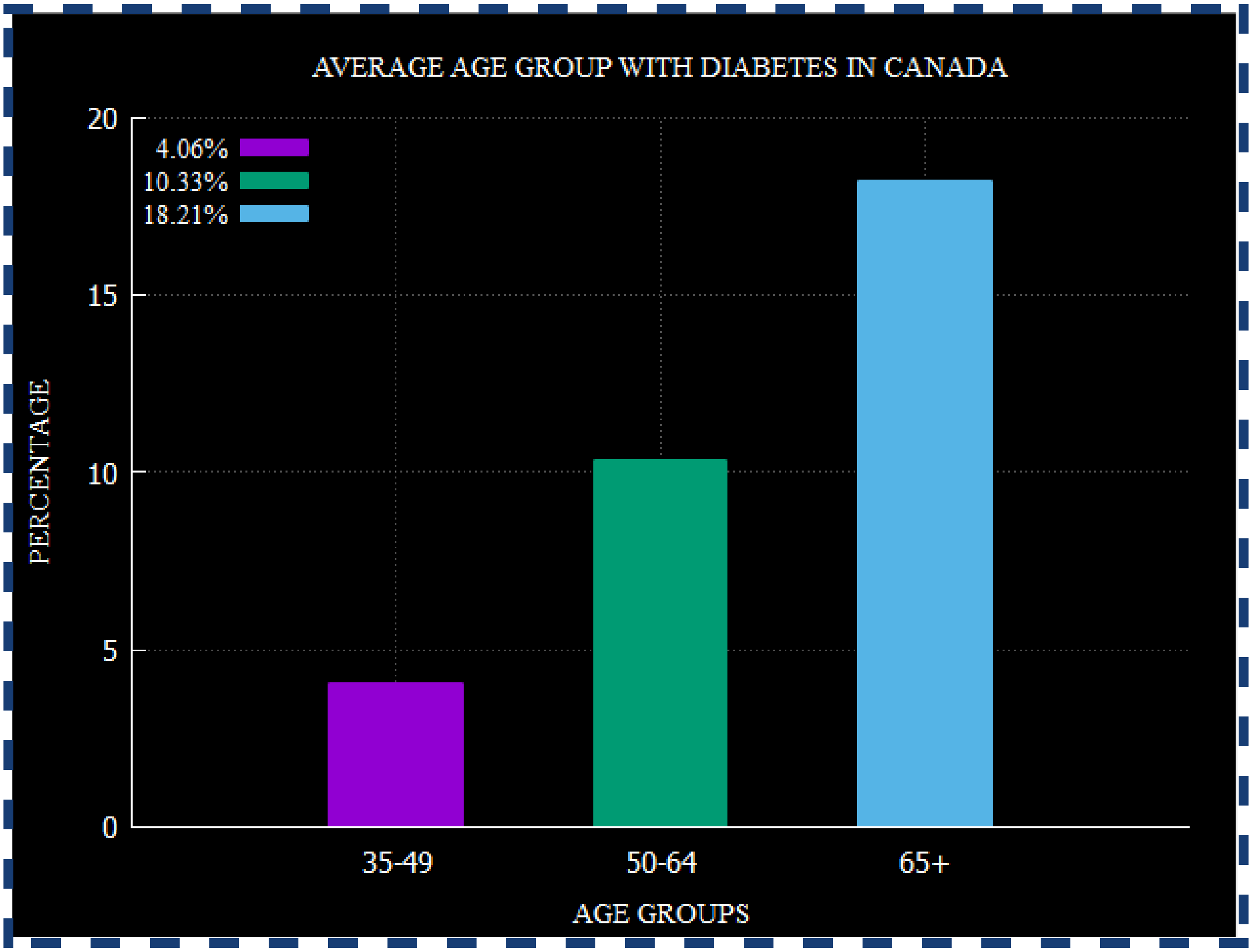


Figure #9 - Percentages per Age Group Displayed in a Bar Graph

From the illustrated graph, it is evident that the percentage of the population with diabetes is most concentrated in the 65+ age group. This correlates with the deduced prevalence of diabetes amongst the elderly population, mainly as a result of insulin resistance that comes with age.

Conclusion

To conclude, the results of the program reveal much about the prevalence of diabetes in Canada over the years. In terms of provinces, Ontario consistently demonstrates the highest diabetes rates, exceeding the national average of 10.87%, while British Columbia maintains the lowest rates, falling below the national average. Despite this, the graphs indicate that British Columbia experienced a notable increase in diabetes prevalence from 2015 to 2021. When analyzing percentages over the years, the data highlights a sharp rise in the number of diagnosed cases in 2019, with the lowest rate occurring in 2016. While extracting age group data, it is apparent that older age groups account for the majority of individuals living with diabetes.

On a national level, the average suggests that **in a sample of 100 individuals, approximately 10 would have diabetes**, emphasizing the condition's substantial impact on the population. By further analyzing this data, it is possible to make informed decisions regarding resource allocation and research investment to better manage diabetes and improve the overall health of the affected population.

Overall, the experience of creating this project was challenging yet equally rewarding, as the skills acquired in C programming and GNU Plot functionalities were effectively showcased to analyze a real world and relevant issue.

One of the biggest challenges faced was at the early stages when determining a method to grab the data from the CSV file. Upon determining that the delimiters used to separate the strings in the file were quotation marks (“”) and commas, they were used accordingly with the **strtok** function to split each line of a CSV file into individual fields, allowing the code to extract specific data from each category (i.e. GEO containing province names) and store it in separate arrays, while skipping unnecessary columns. In doing so, the correct data was extracted which set the way for the rest of the code. Upon further reflecting on the project’s experience, there were a few aspects that could have been improved. One way in which the code may have been optimized is using **structs**. This would eliminate the need for repeatedly defining arrays and variables for national and the respective provinces throughout the program, simplifying the program in terms of legibility and data organization. Additionally, parts of the code had numerous repeated **printf** statements for the output lines that could have been reduced for better optimization.

Despite the challenges and areas of improvements, the team came together to troubleshoot and reflect, thus all was accomplished in a timely and organized manner marking the successful completion of this project.

The source codes (C programming and GNUPlot) can be found below.

C Source Code:

```
#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int
main(void) {
  //Relevant Arrays
  char GEO[300][50];
  char Age_group[300][50];
  char Sex[300][50];
  char Value[300][50];

  double Value_Conv[300];

  char line[2000];
  char REF_DATE[1000][1000];

  //Necessary Pointers
  FILE * fp;
  char * sp; //Used as a way to grab the individual data entries

  //Line variables
  int REF_lines = 0;
  int GEO_lines = 0;
  int AG_lines = 0;
  int SEX_lines = 0;
  int VAL_lines = 0;
  int value_index = 0;
  int line_count = 0;

  //# of data entries
  int data = 210;

  fp = fopen("statscan_diabetes.csv", "r");

  if (fp == NULL) {
    printf("Error: could not open file\n");
    return 1;
  }

  //Read and Grab all the Relevant Data from the .csv File (line_count to prevent truncating, 210 is # of data entries
present in csv)
  while (fgets(line, 2000, fp) != NULL && line_count < data) {
    //Ignore the header columns
    if (strstr(line, "REF_DATE") != NULL) {
      continue;
    }
    //GRAB THE REF_DATES
    sp = strtok(line, ",\""); //take a look at the line and start at the beginning, and stop when it sees a comma or quotation marks as it sees them as delimiters so the function will ignore the quotation marks and treat the data inside as a single entity.
    strcpy(REF_DATE[REF_lines], sp);
    REF_lines++;

    //GRAB THE GEO LOCATIONS
    sp = strtok(NULL, ",\""); //pick up where you left off, after the next comma and grab that chunk of data
    strcpy(GEO[GEO_lines], sp);
    GEO_lines++;

    //SKIP
    sp = strtok(NULL, ",");

    //GRAB AGE
    sp = strtok(NULL, ",\"");
    strcpy(Age_group[AG_lines], sp);
    AG_lines++;

    //GRAB SEX
    sp = strtok(NULL, ",\"");
    strcpy(Sex[SEX_lines], sp);
    SEX_lines++;

    // Skip irrelevant columns
    for (int i = 0; i < 8; i++) {
      sp = strtok(NULL, ",");
    }

    //GRAB Value
    sp = strtok(NULL, ",\"");
    strcpy(Value[VAL_lines], sp);
    VAL_lines++;
    line_count++;
  }

  fclose(fp);

  //Convert String Values from 'Value' to Double Values, Place to 'Value_Conv'
  int i = 0, j = 0;

  for (i = 0; i < VAL_lines; i++) {
    Value_Conv[j] = atof(Value[i]); // convert string to double
    j++;
  }

  //QUESTION #1 (a): Averages of the percentage of the population that are diagnosed with diabetes[PROVINCIAL AVERAGES].

  //Necessary Variables needed to Compute
  double ontario_total = 0.0, quebec_total = 0.0, bc_total = 0.0, alberta_total = 0.0;
  int ontario_count = 0, quebec_count = 0, bc_count = 0, alberta_count = 0;

  // Calculate Provincial Averages
  for (int i = 0; i < VAL_lines; i++) {
    if (strcmp(GEO[i], "Ontario") == 0) {
      if (strcmp(Value[i], "F") != 0) {
        ontario_total += Value_Conv[i];
        ontario_count++;
      }
    } else if (strcmp(GEO[i], "British Columbia") == 0) {
      if (strcmp(Value[i], "F") != 0) {
        bc_total += Value_Conv[i];
        bc_count++;
      }
    } else if (strcmp(GEO[i], "Quebec") == 0) {
```

```
    if (strcmp(Value[i], "F") != 0) {
      quebec_total += Value_Conv[i];
      quebec_count++;
    }
  } else if (strcmp(GEO[i], "Alberta") == 0) {
    if (strcmp(Value[i], "F") != 0) {
      alberta_total += Value_Conv[i];
      alberta_count++;
    }
  }
}

//Calculate Individual Province Averages Based on Count
double p_ontario_avg = ontario_total / ontario_count;
double p_quebec_avg = quebec_total / quebec_count;
double p_bc_avg = bc_total / bc_count;
double p_alberta_avg = alberta_total / alberta_count;

// Output Provincial Averages
printf("\n");
printf("#1 a) \n");
printf("---------------------------\n");
printf("Provincial Averages:\n");
printf("Ontario: %.2f%%\n", p_ontario_avg);
printf("Quebec: %.2f%%\n", p_quebec_avg);
printf("British Columbia: %.2f%%\n", p_bc_avg);
printf("Alberta: %.2f%%\n", p_alberta_avg);
printf("---------------------------\n");
printf("\n");

//QUESTION #1 (b): Averages of the percentage of the population that are diagnosed with diabetes[NATIONAL AVERAGE].

//Necessary Variables needed to Compute
double national_sum = 0.0;
int national_count = 0;

// Calculate National Average
for (int i = 0; i < VAL_lines; i++) {
  if (strcmp(GEO[i], "Canada (excluding territories)") == 0) {
    if (strcmp(Value[i], "F") != 0) {
      national_sum += Value_Conv[i];
      national_count++;
    }
  }
}

//Print out National Average
double p_national_avg = national_sum / national_count;
printf("#1 b) \n");
printf("---------------------------\n");
printf("National Average: %.2f%%\n", p_national_avg);
printf("---------------------------\n");
printf("\n");

//QUESTION #1 (c): Yearly Averages, Average Per Year for each Province and Whole Country.

//Necessary Variables needed to Compute

//row = # of years
double ontario_yearly_totals[7] = {
  0.0
};

int ontario_yearly_counts[7] = {
  0
};
```

```
double quebec_yearly_totals[7] = {
  0.0
};

int quebec_yearly_counts[7] = {
  0
};

double bc_yearly_totals[7] = {
  0.0
};

int bc_yearly_counts[7] = {
  0
};

double alberta_yearly_totals[7] = {
  0.0
};

int alberta_yearly_counts[7] = {
  0
};

double national_yearly_totals[7] = {
  0.0
};

int national_yearly_counts[7] = {
  0
};

double yearly_totals[7] = {
  0.0
};

int yearly_counts[7] = {
  0
};

//Variables for compute
int counter = 0;
double yearly_avg, national_avg, alberta_avg, ontario_avg, quebec_avg,
bc_avg;

//Cycles through each of the years to calculate averages
for (int year = 2015; year <= 2021; year++) {
  //Converts the year to a string to be compared to for each year
  char year_str[5];
  sprintf(year_str, "%d", year);
  int index = year - 2015;

  for (int i = 0; i < VAL_lines; i++) {
    // Calculate Yearly Average in the current year, and National &Province Averages in the current year
    if (strcmp(REF_DATE[i], year_str) == 0 && strcmp(Value[i], "F") != 0) {
      if (strcmp(GEO[i], "Alberta") == 0) {
        alberta_yearly_totals[index] += Value_Conv[i];
        alberta_yearly_counts[index]++;
      }

      if (strcmp(GEO[i], "Ontario") == 0) {
        ontario_yearly_totals[index] += Value_Conv[i];
        ontario_yearly_counts[index]++;
      }

      if (strcmp(GEO[i], "Quebec") == 0) {
        quebec_yearly_totals[index] += Value_Conv[i];
```

```
          quebec_yearly_counts[index]++;
        }

        if (strcmp(GEO[i], "British Columbia") == 0) {
          bc_yearly_totals[index] += Value_Conv[i];
          bc_yearly_counts[index]++;
        }

        if (strcmp(GEO[i], "Canada (excluding territories)") == 0) {
          national_yearly_totals[index] += Value_Conv[i];
          national_yearly_counts[index]++;
        }

        yearly_totals[index] += Value_Conv[i];
        yearly_counts[index]++;
      }
    }
  }

  //Print out results[TABLE]
  printf("#1 c) \n");
  printf("------------------------------------------------------------------------------------ \n");
  printf("\t   Averages of People (All Ages) that have Diabetes in Canada (in %%)\n");
  printf("------------------------------------------------------------------------------------ \n");
  printf(" Region:   |2015|     |2016|     |2017|     |2018|     |2019|     |2020|     |2021|\n");

  //Print out results for each province
  printf("   AB ");
  for (int i = 0; i < 7; i++) {
    printf("%10.2lf%%", alberta_yearly_totals[i] / alberta_yearly_counts[i]);
  }

  printf("\n");

  printf("   BC ");
  for (int i = 0; i < 7; i++) {
    printf("%10.2lf%%", bc_yearly_totals[i] / bc_yearly_counts[i]);
  }

  printf("\n");

  printf("   ON ");
  for (int i = 0; i < 7; i++) {
    printf("%10.2lf%%", ontario_yearly_totals[i] / ontario_yearly_counts[i]);
  }

  printf("\n");

  printf("   QC ");
  for (int i = 0; i < 7; i++) {
    printf("%10.2lf%%", quebec_yearly_totals[i] / quebec_yearly_counts[i]);
  }

  printf("\n");

  printf("   CA ");
  for (int i = 0; i < 7; i++) {
    printf("%10.2lf%%", national_yearly_totals[i] / national_yearly_counts[i]);
  }

  printf("\n");
  printf("------------------------------------------------------------------------------------ \n");

  //QUESTION #1 (d) The avg % of diabetes among age groups (35-49, 60-64, 65+). One average per age group (all years)
for each province and the whole country.

  //Row = province, col = age groups
```

```
double yearly_age_totals[5][3] = {
  0
};

int yearly_age_counts[5][3] = {
  0
};

for (int year = 2015; year <= 2021; year++) {
  char year_str[5];
  sprintf(year_str, "%d", year);

  for (int i = 0; i < VAL_lines; i++) {
    if (strcmp(REF_DATE[i], year_str) == 0 && strcmp(Value[i], "F") != 0) {
      int age_group = -1;
      //If statements to seperate the Age Groups
      if (strcmp(Age_group[i], "35 to 49 years") == 0) {
        age_group = 0;
      } else if (strcmp(Age_group[i], "50 to 64 years") == 0) {
        age_group = 1;
      } else if (strcmp(Age_group[i], "65 years and over") == 0) {
        age_group = 2;
      }

      //Split the Averages
      if (age_group == 0) //35-49
      {
        if (strcmp(GEO[i], "Alberta") == 0) {
          yearly_age_totals[0][0] += Value_Conv[i];
          yearly_age_counts[0][0]++;
        }

        if (strcmp(GEO[i], "Ontario") == 0) {
          yearly_age_totals[1][0] += Value_Conv[i];
          yearly_age_counts[1][0]++;
        }

        if (strcmp(GEO[i], "Quebec") == 0) {
          yearly_age_totals[2][0] += Value_Conv[i];
          yearly_age_counts[2][0]++;
        }

        if (strcmp(GEO[i], "British Columbia") == 0) {
          yearly_age_totals[3][0] += Value_Conv[i];
          yearly_age_counts[3][0]++;
        }

        if (strcmp(GEO[i], "Canada (excluding territories)") == 0) {
          yearly_age_totals[4][0] += Value_Conv[i];
          yearly_age_counts[4][0]++;
        }
      }

      //Split the Averages
      if (age_group == 1) //50-64
      {
        if (strcmp(GEO[i], "Alberta") == 0) {
          yearly_age_totals[0][1] += Value_Conv[i];
          yearly_age_counts[0][1]++;
        }

        if (strcmp(GEO[i], "Ontario") == 0) {
          yearly_age_totals[1][1] += Value_Conv[i];
          yearly_age_counts[1][1]++;
        }

        if (strcmp(GEO[i], "Quebec") == 0) {
            yearly_age_totals[2][1] += Value_Conv[i];
            yearly_age_counts[2][1]++;
          }

          if (strcmp(GEO[i], "British Columbia") == 0) {
            yearly_age_totals[3][1] += Value_Conv[i];
            yearly_age_counts[3][1]++;
          }

          if (strcmp(GEO[i], "Canada (excluding territories)") == 0) {
            yearly_age_totals[4][1] += Value_Conv[i];
            yearly_age_counts[4][1]++;
          }
        }

        //Split the Averages
        if (age_group == 2) //65+
        {
          if (strcmp(GEO[i], "Alberta") == 0) {
            yearly_age_totals[0][2] += Value_Conv[i];
            yearly_age_counts[0][2]++;
          }

          if (strcmp(GEO[i], "Ontario") == 0) {
            yearly_age_totals[1][2] += Value_Conv[i];
            yearly_age_counts[1][2]++;
          }

          if (strcmp(GEO[i], "Quebec") == 0) {
            yearly_age_totals[2][2] += Value_Conv[i];
            yearly_age_counts[2][2]++;
          }

          if (strcmp(GEO[i], "British Columbia") == 0) {
            yearly_age_totals[3][2] += Value_Conv[i];
            yearly_age_counts[3][2]++;
          }

          if (strcmp(GEO[i], "Canada (excluding territories)") == 0) {
            yearly_age_totals[4][2] += Value_Conv[i];
            yearly_age_counts[4][2]++;
          }
        }
      }
    }
  }

  printf("\n");
  printf("#1 d)\n");

  // Print out the results
  printf("------------------------------------------------------------------- \n");
  printf("\tAverages of People with Diabetes in Canada (in %%)\n");
  printf("------------------------------------------------------------------- \n");
  printf("Age Group:    |AB|       |ON|       |QC|       |BC|       |CA|\n");

  // Print out results for each age group and region
  int region = 0;
  printf("  35-49 ");
  for (region = 0; region < 5; region++) {
    printf("%10.2lf%%", yearly_age_totals[region][0] / yearly_age_counts[region][0]);
  }
  printf("\n");

  printf("  50-64 ");
  for (region = 0; region < 5; region++) {
    printf("%10.2lf%%", yearly_age_totals[region][1] / yearly_age_counts[region][1]);
```

```
}
printf("\n");

printf("    65+ ");
for (region = 0; region < 5; region++) {
  printf("%10.2lf%%", yearly_age_totals[region][2] / yearly_age_counts[region][2]);
}
printf("\n");
printf("-------------------------------------------------------------------- \n");
printf("\n");

//#2 Province that has highest percentage of diabetes and which province has the lowest percentage.
printf("#2\n");
double province_avgs[4] = {
  0.0
};

char * province_names[5] = {
  "Ontario",
  "Quebec",
  "British Columbia",
  "Alberta"
};

province_avgs[0] = p_ontario_avg;
province_avgs[1] = p_quebec_avg;
province_avgs[2] = p_bc_avg;
province_avgs[3] = p_alberta_avg;

int province_avgs_size = sizeof(province_avgs) / sizeof(province_avgs[0]);
int max = 0, min = 0;
for (int i = 1; i < province_avgs_size; i++) {
  if (province_avgs[i] < province_avgs[min]) {
    min = i;
  }

  if (province_avgs[i] > province_avgs[max]) {
    max = i;
  }
}

printf("-------------------------------------------------------------------------\n");
printf("➪ Province that has Highest %% of Diabetes is %s with %.2lf%% \n", province_names[max], province_avgs[max]);
printf("➪ Province that has Lowest %% of Diabetes is %s with %.2lf%% \n", province_names[min], province_avgs[min]);
printf("-------------------------------------------------------------------------\n");
printf("\n");

//#3 Provinces that have diabetes percentages above/below the national average
printf("#3\n");
printf("-------------------------------------------------------------------------\n");

// Compare each province's diabetes percentage to the national average
for (int i = 0; i < province_avgs_size; i++) {
  if (province_avgs[i] > p_national_avg) {
    printf("➪ %s has diabetes percentage above the national average\n", province_names[i]);
    printf("-------------------------------------------------------------------------\n");
  } else {
    printf("➪ %s has diabetes percentage below the national average\n", province_names[i]);
  }
}

printf("-------------------------------------------------------------------------\n");

//#4 Year and Province that has the Highest/Lowest % of Diabetes

// Determine year with highest diabetes rate
double max_yearly_avg = -1.0; //set max_yearly_avg to a negative number so it can be properly compared
```

```c
int max_year = -1;

for (int i = 0; i < 7; i++) {
  yearly_avg = yearly_totals[i] / yearly_counts[i];
  if (yearly_avg > max_yearly_avg) {
    max_yearly_avg = yearly_avg;
    max_year = i + 2015;
  }
}

// Determine province with highest diabetes rate
double max_province_avg = -1.0;
char * max_province;

//Tallys the total to then average out
for (int i = 0; i < 7; i++) {
  alberta_avg += alberta_yearly_totals[i] / alberta_yearly_counts[i];
  ontario_avg += ontario_yearly_totals[i] / ontario_yearly_counts[i];
  quebec_avg += quebec_yearly_totals[i] / quebec_yearly_counts[i];
  bc_avg += bc_yearly_totals[i] / bc_yearly_counts[i];

  //Average of each province
  alberta_avg /= 7;
  ontario_avg /= 7;
  quebec_avg /= 7;
  bc_avg /= 7;

  if (alberta_avg > max_province_avg) {
    max_province_avg = alberta_avg;
    max_province = "Alberta";
  }

  if (ontario_avg > max_province_avg) {
    max_province_avg = ontario_avg;
    max_province = "Ontario";
  }

  if (quebec_avg > max_province_avg) {
    max_province_avg = quebec_avg;
    max_province = "Quebec";
  }

  if (bc_avg > max_province_avg) {
    max_province_avg = bc_avg;
    max_province = "British Columbia";
  }
}

printf("\n");
printf("#4 \n");
printf("------------------------------------------------------------------------\n");
printf("⇨ Province with highest diabetes rate: %s\n", max_province);
printf("⇨ Year with highest diabetes rate: %d\n", max_year);
printf("------------------------------------------------------------------------\n");

// Determine year with lowest diabetes rate
double min_yearly_avg = 50.0; //Set min_yearly_avg to a high value so it can the compared correctly
int min_year = -1;

for (int i = 0; i < 7; i++) {
  yearly_avg = yearly_totals[i] / yearly_counts[i];
  if (yearly_avg < min_yearly_avg) {
    min_yearly_avg = yearly_avg;
    min_year = i + 2015;
  }
}
```

```
  // Determine province with lowest diabetes rate
  double min_province_avg = 50.0;
  char * min_province;

  //Tallys the total to then average out
  for (int i = 0; i < 7; i++) {
    alberta_avg += alberta_yearly_totals[i] / alberta_yearly_counts[i];
    ontario_avg += ontario_yearly_totals[i] / ontario_yearly_counts[i];
    quebec_avg += quebec_yearly_totals[i] / quebec_yearly_counts[i];
    bc_avg += bc_yearly_totals[i] / bc_yearly_counts[i];

    //Averages of each Province
    alberta_avg /= 7;
    ontario_avg /= 7;
    quebec_avg /= 7;
    bc_avg /= 7;

    if (alberta_avg < min_province_avg) {
      min_province_avg = alberta_avg;
      min_province = "Alberta";
    }

    if (ontario_avg < min_province_avg) {
      min_province_avg = ontario_avg;
      min_province = "Ontario";
    }

    if (quebec_avg < min_province_avg) {
      min_province_avg = quebec_avg;
      min_province = "Quebec";
    }

    if (bc_avg < min_province_avg) {
      min_province_avg = bc_avg;
      min_province = "British Columbia";
    }
  }

  printf("-----------------------------------------------------------------------\n");
  printf("➪ Province with lowest diabetes rate: %s\n", min_province);
  printf("➪ Year with lowest diabetes rate: %d\n", min_year);
  printf("-----------------------------------------------------------------------\n");
  printf("\n");

  return 0;
}
```

GNUPlot Source Code:

```
//Q(5) graph of the diabetes percentages for the years 2015 to 2021 (all age groups and genders together) for the four
provinces and the national average (indicated as Canada excluding territories)

//Book1.csv to extract data for the plot itself
2015,10.767,9.32,10.9,9.3,10.6
2016,12.2,9.77,9.82,8.53,10.7
2017,11.983,11.967,9.583,10.14,10.95
2018,11.28,11.02,10.65,8.52,10.78
2019,13.03,11.33,10.483,11.44,11.7
2020,11.17,12.88,11.42,9.04,10.6
2021,11.48,9.82,10.47,11.65,10.75

//Setting titles and axis labels
set datafile separator ','
set title 'YEARLY DIABETES AVERAGES FROM 2015-2021'
set xlabel 'YEARS'
set ylabel 'PERCENTAGE'
set xrange [2015:2021]
set yrange [5:15]
set grid
set title font "Times New Roman"
set xlabel font "Times New Roman"
set ylabel font "Times New Roman"
set key font "Times New Roman"
set border 3
set tics nomirror

//Setting the colours for the labels, titles, borders, etc.
set obj 1 rect from screen 0,0 to screen 1,1 fillcolor rgb"black" behind
set tics tc rgb"white"
set ylabel tc rgb"white"
set xlabel tc rgb"white"
set title tc rgb"white"
set key tc rgb"white"
set key bottom right
set border 3 lc"white"

//Plotting the graph
plot 'Book1.csv' u 1:2 w lp lw 1.5 lc 7 lt 7 title 'Ontario','Book1.csv' u 1:3 w lp lw 1.5 lc 2 lt 7 title
'Alberta','Book1.csv' u 1:4 w lp lw 1.5 lc 0 lt 7 title 'Quebec','Book1.csv' u 1:5 w lp lw 1.5 lc 17 lt 7 title
'BC','Book1.csv' u 1:6 w lp lw 1.5 lc 3 lt 0 title 'Canada (excluding territories)'



//Q(6) Bar Chart that shows the average percentages of diabetes among the three age groups for the entire country
(Canada excluding territories)

//Book2.csv file to extract data
1,4.0642,,,35-49
2,,10.32857,,50-64
3,,,18.214,65+

//Set labels titles for bar graph
set datafile separator ','
set yrange [0:20]
set xrange [0:4]
set style fill solid
set boxwidth 0.5
set title 'AVERAGE AGE GROUP WITH DIABETES IN CANADA'
set xlabel 'AGE GROUPS'
set ylabel 'PERCENTAGE'
set title font "Times New Roman"
set xlabel font "Times New Roman"
set ylabel font "Times New Roman"
set key font "Times New Roman"
set border 3
set tics nomirror
set grid
set key top left

set obj 1 rect from screen 0,0 to screen 1,1 fillcolor rgb"black" behind
set tics tc rgb"white"
set ylabel tc rgb"white"
set xlabel tc rgb"white"
set title tc rgb"white"
set key tc rgb"white"
set border 3 lc"white"

//Plot the bar graph
plot "Book2.csv" using 1:2:xtic(5) title "4.06%" with boxes, "Book2.csv" using 1:3:xtic(5) title "10.33%" with boxes,
"Book2.csv" using 1:4:xtic(5) title "18.21%" with boxes
```