Course: ESNF 614 – Fall 2022
Lab: #1
Student Name: Khoi Nguyen
ID: 10171399
Submission Date: September 20, 2022

Exercise B:

Code:

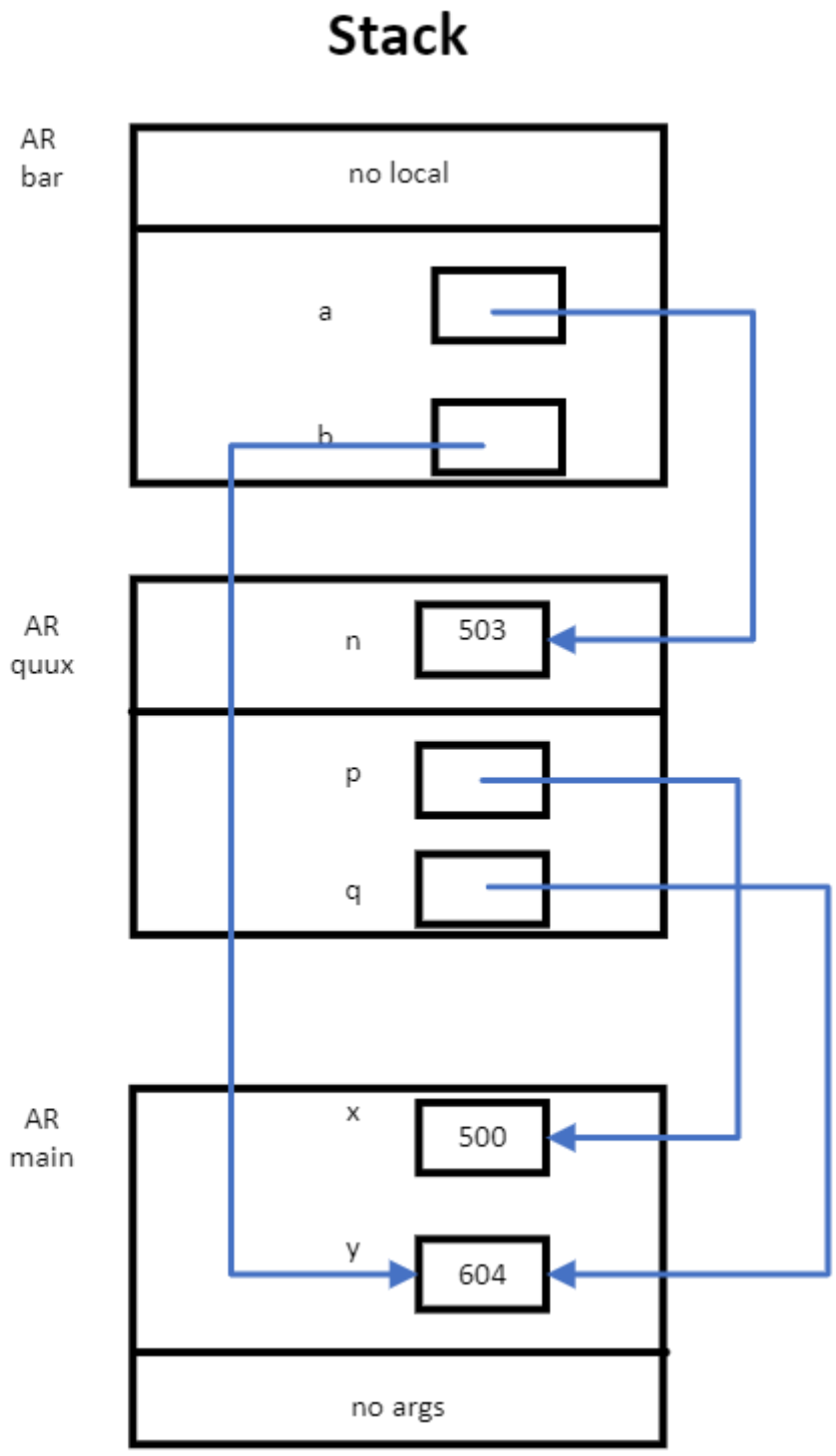
```
/*
 *  lab1exe_B.cpp
 *  ENSF 614 Lab 1, exercise B
 *  Created by Mahmood Moussavi
 *  Completed by: Khoi Nguyen
 */

#include <iostream>
#include <cmath>
using namespace std;

const double G = 9.8;   /* gravitation acceleration 9.8 m/s^2 */
const double PI = 3.141592654;

void create_table(double v);
double Projectile_travel_time(double a, double v);
double Projectile_travel_distance(double a, double v);
double degree_to_radian(double d);

int main(void)
{
    int n;
    double velocity;

    cout << "Please enter the velocity at which the projectile is launched
(m/sec): ";
    cin >> velocity;

    if(!cin)  // means if cin failed to read
    {
        cout << "Invlid input. Bye...\n";
        exit(1);
    }

    while (velocity < 0 )
    {
        cout << "\nplease enter a positive number for velocity: ";
        cin >> velocity;
        if(!cin)
        {
            cout << "Invalid input. Bye...";
            exit(1);
```

```
        }
    }

    create_table(velocity);

    return 0;
}

//Converting degree value to radian
double degree_to_radian(double degree)
{
    return degree * PI/180;
}

//Calculating the projectile travel time using angle and initial velocity
double Projectile_travel_time(double angle, double v)
{
    return 2*v*sin(angle)/G;
}

//Calculating the projectile travel distance using angle and initial velocity
double Projectile_travel_distance(double angle, double v)
{
    return pow(v,2)*sin(2*angle)/G;
}

//Printing a table containing the angle of launch and corresponding time and
distance travelled associated with the initial velocity
void create_table (double v)
{
    printf("Angle (deg) \tt (sec) \td (m)\n");
    double angle;
    double distance;
    double time;
    for (int degree = 0; degree <= 90; degree+=5)
    {
        angle = degree_to_radian(degree);
        time = Projectile_travel_time(angle, v);
        distance = Projectile_travel_distance(angle, v);
        printf("%i\t%f\t%f\n", degree, time, distance);
    }
}
```

Output:

```
$ ./lab1exe_B
Please enter the velocity at which the projectile is launched (m/sec): 100
Angle (deg)     t (sec)         d (m)
0       0.000000        0.000000
5       1.778689        177.192018
10      3.543840        349.000146
15      5.282021        510.204082
20      6.980003        655.905724
25      8.624862        781.678003
30      10.204082       883.699392
35      11.705642       958.870021
40      13.118114       1004.905870
45      14.430751       1020.408163
50      15.633560       1004.905870
55      16.717389       958.870021
60      17.673988       883.699391
65      18.496077       781.678003
70      19.177400       655.905724
75      19.712772       510.204081
80      20.098117       349.000146
85      20.330504       177.192018
90      20.408163       -0.000000
```

Exercise D2


Stack
AR
bar
no local
a
b
AR
quux
n
503
p
q
AR
main
x
500
y
604
no args

Exercise E

```
/*
 *  lab1exe_E.cpp
 *  ENSF 619 Lab 1 Exercise E1
 *  Completed by: Khoi Nguyen
 */

#include <iostream>
using namespace std;

void time_convert(int ms_time, int *minutes_ptr, double *seconds_ptr);
/*
 * Converts time in milliseconds to time in minutes and seconds.
 * For example, converts 123400 ms to 2 minutes and 3.4 seconds.
 * REQUIRES:
 *    ms_time >= 0.
 *    minutes_ptr and seconds_ptr point to variables.
 * PROMISES:
 *    0 <= *seconds_ptr & *seconds_ptr < 60.0
 *    *minutes_ptr minutes + *seconds_ptr seconds is equivalent to
 *    ms_time ms.
 */

int main(void)
{
  int millisec;
  int minutes;
  double seconds;
  int nscan;

  cout << "Enter a time interval as an integer number of milliseconds: ";

 // printf("Enter a time interval as an integer number of milliseconds: ");
  cin >> millisec;

  if (!cin) {
    cout << "Unable to convert your input to an int.\n";
    exit(1);
  }

  if (millisec < 0) {
    cout << "Time cannot be negative.\n";
```

```
    exit(1);
  }

  cout << "Doing conversion for input of " <<  millisec <<" milliseconds ... \n",
millisec;

  /* MAKE A CALL TO time_convert HERE. */
  time_convert(millisec, &minutes, &seconds);
  cout << "That is equivalent to " << minutes << " minute(s) and " << seconds <<
" second(s).\n";
  return 0;
}

/* PUT YOUR FUNCTION DEFINITION FOR time_convert HERE. */
void time_convert(int ms_time, int *minutes_ptr, double *seconds_ptr)
{
  *minutes_ptr = (int) ms_time/60000; // converting millisec to the nearest
minutes
  *seconds_ptr = (double) ms_time/1000 - *minutes_ptr*60; //getting the remaining
seconds
}
```

Output:

```
$ ./lab1exe_E
Enter a time interval as an integer number of milliseconds: 123400
Doing conversion for input of 123400 milliseconds ...
That is equivalent to 2 minute(s) and 3.4 second(s).
```