BIL105E – Introduction to Scientific and Engineering Computing (C)

Spring 2015 - 2016

CRN: 21837

Homework 1

Assignment Subject: ***Writing a C program which stimulates a ball catching play between two teams.***

Assignment Due Date: ***09.03.2016 23:00***

Submission Date: ***09.03.2016***

Student Name: ***Korel Chairoula***

Student ID: ***150140910***

e-mail: ***korel_hayrullah@hotmail.com***

Introduction Section

First of all, this project was really fun to work with it. Simply, the project was about a ball catching play between two teams. The program at the beginning takes some information from the user such as: Player per teams, probability for successful passes, successful passes to win a single round and a target score. After that the program takes this information and starts the game. The written code takes this information and processes it. Finally, when one team reaches the target score entered from the user, the team wins. If players in the same team cannot make successful passes, the ball passes to the other team and the same goes on and on until one team reaches the passes that the user entered in. Of course there is a probability to pass the ball between players. If the players in the same team can reach the passes that the user entered, this team gets 1 point. The winning limit is up to the user. It is a program. The user can select whatever he/she wants in that game to be happen. Here is an example of the program.

```
Korel-MacBook-Pro:Desktop Korel.Hayrullah$ ./a.out
Enter the number of players per team (N)
6
Enter the probability of succesful pass (P)
60
Enter the number of passes to win a single round (W)
3
Enter target score (S)
2
Round-1:
        Team2 is selected.
        Player5
        Team 1 captured the ball!
        Player2
        Team2 captured the ball!
        Player5
        Team 1 captured the ball!
        Player4 -> Player2-> Player3
        Success! The new score of Team1 is 1

Round-2:
        Team1 is selected.
        Player6 -> Player5-> Player4
        Success! The new score of Team1 is 2

GAME OVER: Team1 reached the target score (2) and won the game.
```

Development Environment

Operating System : ***OSX El Capitan***
Programming Language : ***C***
Compiler : ***GCC***
Source File*: My program includes one file c named* ***150140910.c***
Header Files: *The header files are* ***<stdio.h>****,* ***<stdlib.h>*** *and* ***<time.h>****.*

I used header files <stdio.h> because it is the default library in order to run default functions. Like: “printf”, “scanf” and etc. Secondly I used <stdlib.h> because the project that I worked on needed a function for generating random number. Normally, “random();” is there for it, however, every time when it will use random() function it will start from the same point to give numbers. In the library <stlib.h>, there is a function called “srand();” which will change the initiating point, but only this does not give me the solution to create random numbers. Another library called <time.h> will solve the problem using it like “srand(time(NULL));”. Now, this will create random number because the time function changes every second and the initiating point will change and give us random number which in this project we all needed. After finishing coding my program I used terminal on my MacBook to compile it via gcc. My command was like “Korel-MacBook-Pro:~ Korel.Hayrullah$ gcc 150140910.c”. In order to run it I used in terminal “./a.out” to run my program. There is a easier way to compile and simultaneously compile the program and change the name is “gcc 150140910.c –o MyFirstAssignmentC” and then again to run it “./MyFirstAssignmentC”. Here are some screenshots to explain this via image and in the meantime my program after run.

```
Last login: Tue Mar  8 22:10:58 on ttys000
Korel-MacBook-Pro:~ Korel.Hayrullah$ cd Desktop
Korel-MacBook-Pro:Desktop Korel.Hayrullah$ gcc 150140910.c
Korel-MacBook-Pro:Desktop Korel.Hayrullah$ ./a.out
Enter the number of players per team (N)
4
Enter the probability of succesful pass (P)
80
Enter the number of passes to win a single round (W)
3
Enter target score (S)
2
Round-1:
        Team1 is selected.
        Player1
        Team2 captured the ball!
        Player1 -> Player4-> Player3
        Success! The new score of Team2 is 1

Round-2:
        Team2 is selected.
        Player4
        Team 1 captured the ball!
        Player2
        Team2 captured the ball!
        Player1 -> Player2-> Player3
        Success! The new score of Team2 is 2

GAME OVER: Team2 reached the target score (2) and won the game.
Korel-MacBook-Pro:Desktop Korel.Hayrullah$ _
```

```
Korel-MacBook-Pro:Desktop Korel.Hayrullah$ gcc 150140910.c -o MyFirstAssignmentC
Korel-MacBook-Pro:Desktop Korel.Hayrullah$ ./MyFirstAssignmentC
Enter the number of players per team (N)
3
Enter the probability of succesful pass (P)
90
Enter the number of passes to win a single round (W)
2
Enter target score (S)
2
Round-1:
        Team2 is selected.
        Player3
        Team 1 captured the ball!
        Player2 -> Player1
        Success! The new score of Team1 is 1

Round-2:
        Team2 is selected.
        Player1 -> Player2
        Success! The new score of Team2 is 1

Round-3:
        Team2 is selected.
        Player2 -> Player1
        Success! The new score of Team2 is 2

GAME OVER: Team2 reached the target score (2) and won the game.
Korel-MacBook-Pro:Desktop Korel.Hayrullah$
```

Data Structure and Variables

In my program I used while, for, if, srand, printf, scanf, #include, break, return, int, time and rand. First of all, **int** stands for integer numbers. **Break;** is for “break the loop” in short, to stop the loop and move to the next process. **Srand()** is from the library <stdlib.h> and changes the initialization point of function rand(), however, only srand function is not enough. We have **time()** function for continuously changing the initialization point which is from the <time.h> library. **Rand()** function generates random numbers, but the function rand() without any modification gives every time the same random numbers. **#include** includes the libraries stated above in C programming language. **Printf** function prints everything that is entered in it and **scanf** function gets input information from the user. **If** checks the condition if it is true or not. **While** and **for** are loops which loops again and again until the condition does not provide. I used all of these stated above. The entry of the program follows with some outputs and inputs. After taking the information from the user it starts the “ball catching game”.

int players; //stands for the numbers of players per team which are entered from the user

int probability; //stands for the probability of successful pass rates between players in the same team which is also entered by the user

int passes; //stands for the passes entered by the user to win a single round

int countPasses; //counts the successful passes every time and when the team fails to reach the target pass entered by the use in my program I assigned it to one (1) to be sure it counts correctly

int targetScore; //stands for the target score that each team must reach in order to win the game and it is set from the user

int randomTeam; // this variable picks randomly which teams will start the game and it is limited by "1 + random() % 2" because the projects wants 2 teams

int randomPlayer1; //generates random player number

int randomPlayer2; //generates random player number (these two variables: int randomPlayer1; and int randomPlayer2; may seem same, but they have a tricky difference. Both of these variables gives random player numbers again limited between 1 and the entered number by the user which is player number per team. When passes happen in same team the program gives a random player number and if the successful pass happens it will print it on screen. For example, in the same team there can not be two (2) player ones that pass to each other. For that reason, in the program when it will print the second player it will check it if the random number is same with the first one or not. When it is the same it will generate random numbers until it gives a different number. The loop while checks it until the 2 player numbers are different.

int randomProbability; // There are two different probabilities stated as variables as stated above int probability is the probability rate for successful passes entered by the user. The randomProbability here generates a random probability rate between 1 and 100. To illustrate, "int randomProbability = 1 + rand() % 100". These two variables are checked whether the probability rate entered from the user is greater than or equal to randomProbability. To exemplify, assume that the user entered a probability rate 60. Assume that the machine gave randomProbability rate 55. 60 >=55 this statement is correct and for that reason it will accepted as a successful pass and the player will be able to pass tothe other player.

int Team1 = 0; //this is in order to count Team1's score.

int Team2 = 0; //this is in order to count Team2's score. (If one of these two "Team1 and Team2" reaches the score the user entered in. The game will be finished and the winner will be announced.
Here are some visualizations:

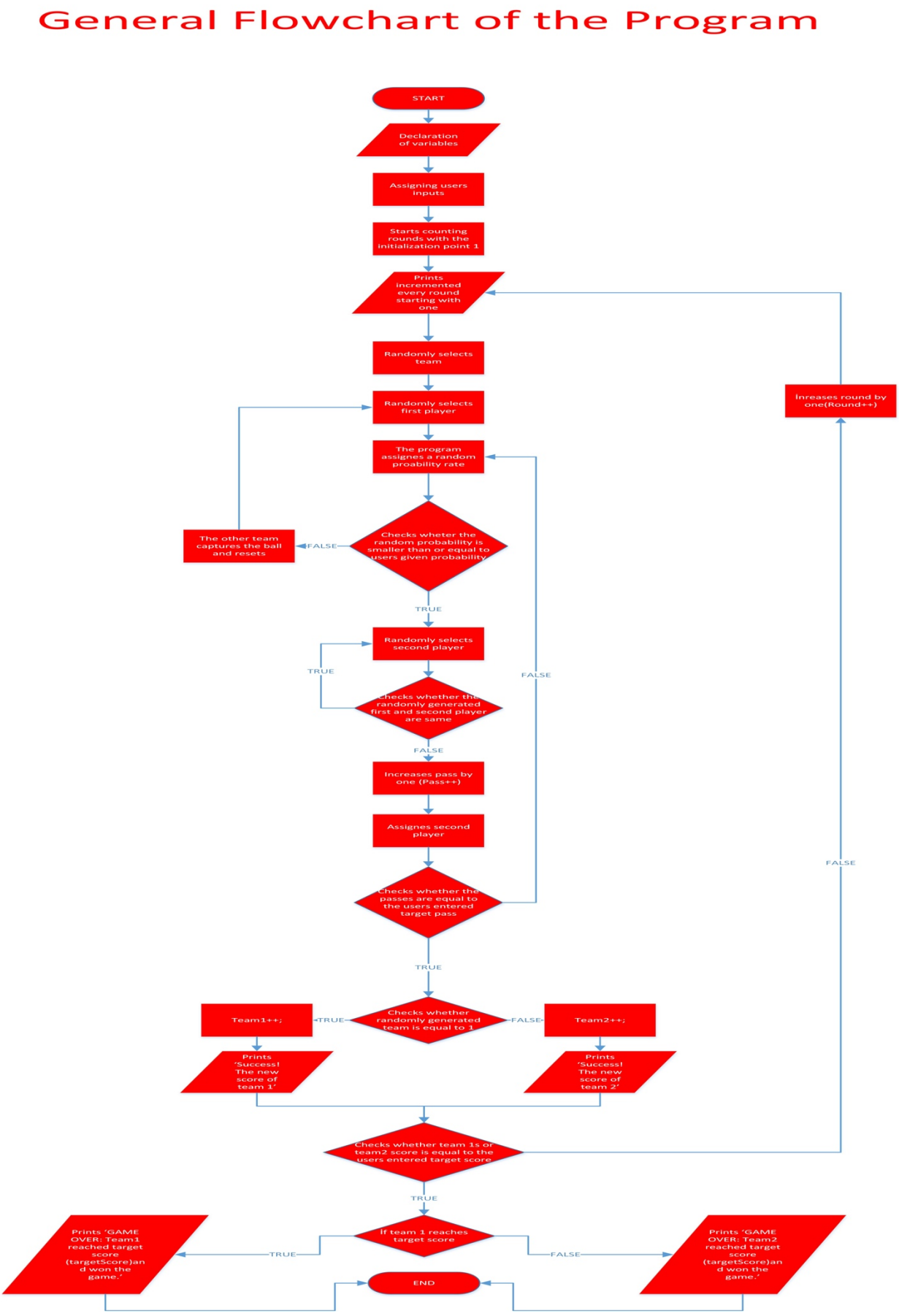
```
/*all the variables that used in the program*/
int players, probability, passes, countPasses, targetScore, randomTeam, randomPlayer1, randomPlayer2, randomProbability;
int Team1 = 0;
int Team2 = 0;
```

```
randomTeam = 1 + rand() % 2; /*generates random team number which is going to start*/

printf("\tTeam%d is selected.\n",randomTeam);

randomPlayer1 = 1 + rand() % players;/*generates random player number*/

printf("\tPlayer%d ", randomPlayer1);
```

```
      while(randomPlayer1 == randomPlayer2){
        randomPlayer2 = 1 + rand() %players;
      }
      randomPlayer1 = randomPlayer2;

      printf("-> Player%d",randomPlayer2);
```

```
while(countPasses < passes){
  randomProbability = 1 + rand() % 100; /*generates random probability rate from 1 to 100*/
  /*the statement "if" below checks the probability rate entered by the user between random generated probability*/
  if(randomProbability <= probability){
    countPasses++;
```

Program Flow



Conclusion

In conclusion, the project overall was a bit challenging for me because it is my first time really writing a program, a program such in a difficult programming language like C. Everyone said that C as a programming language is a challenging language that I will face up through my computer engineering license. At the very beginning it is really hard to understand the working principle, the mentality as every other things in life. Practicing is the key to success. The project, my first assignment gave me a lot of things. A lot of information about C programming language, but for now. I know there are a lot more things to gain experience. Using nested loops gave me the comprehensive understanding of the working principle. To be honest, I gave a lot of tries to work my program correctly. I had some mistakes while making nested loops. Sometimes I could not realize my mistakes. While running the program I had some trouble with loops. Most of the time, it made infinite loops. The other equations and calculations that I made were right. Not, all of them. Most of the time I had trouble regulating the loops. All in all, the project contributed to understand how it works and responses.