CMPE 150 ASSIGNMENT 2 (EMRE GİRGİN)

1-) Problem Description:

In this assignment we have to write a Java program for MARS ,which interviews with the user. Therefore, it should be interactive. Program should not continue when required conditions are not satisfied. For example, if user does not have a university degree, program should not ask him to his field.

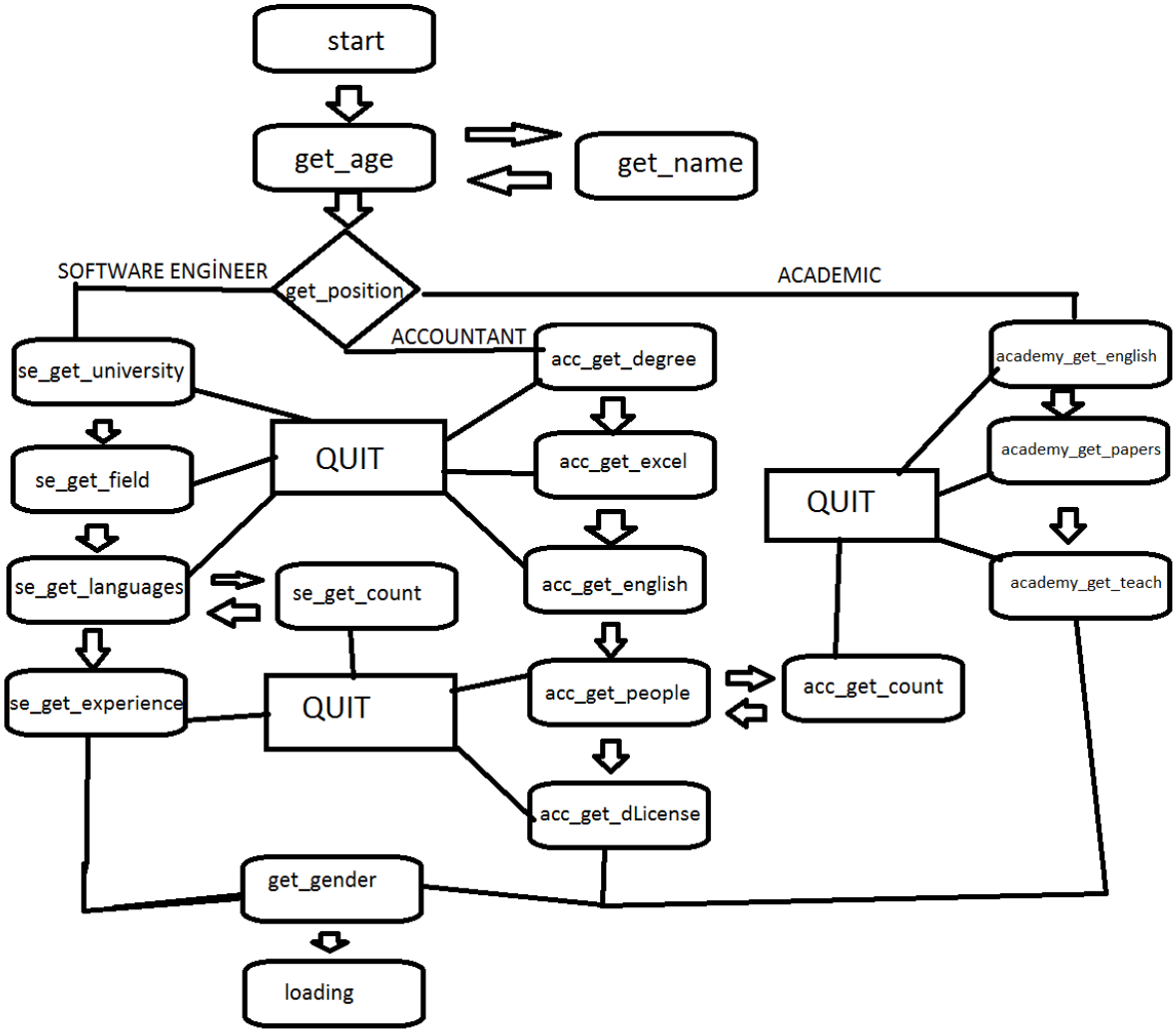
2-) Problem Solution:

In this part I'm going to explain my code by flow charts and screenshots.

*EG2016400099.java

```
 1 import java.util.Scanner;
 2
 3 public class EG2016400099{
 4     public static void start() { 
16     //-----IN COMMON-----
18      * This part is in common for all positions.
31     public static String get_name(Scanner sc) { 
41     public static String get_age(Scanner sc) {
64     public static String get_position(Scanner sc, String name) {
106    //-----SOFTWARE ENGINEER-----
108     * "se_" part of the methods' name represents "sOFTWARE eNGINEER". 
117    public static String se_get_university(Scanner sc, String name) {
138    public static String se_get_field(Scanner sc, String name) {
158    public static String se_get_languages(Scanner sc, String name) {
186        public static String se_get_count(Scanner sc) {
211    public static String se_get_experience(Scanner sc, String name) {
253    //-----ACCOUNTANT-----
255     * "acc_" part of the methods' name represents "accOUNTANT".
261    public static String acc_get_degree(Scanner sc, String name) {
279    public static String acc_get_excel(Scanner sc, String name) {
297    public static String acc_get_english(Scanner sc, String name) {
331    public static String acc_get_people(Scanner sc, String name) {
348        public static String acc_get_count(Scanner sc) {
371    public static String acc_get_dLicense(Scanner sc, String name) {
390    //-----ACADEMIC-----
392     * "academy_" part of the methods' name represents "ACADEMIC".
398    public static String academy_get_english(Scanner sc, String name) {
416    public static String academy_get_papers(Scanner sc, String name) {
436    public static String academy_get_teach(Scanner sc, String name) {
455    //-----GENERAL-----
457     * These three methods are in common for all positions.
461    public static String get_gender(Scanner sc, String name) {
486    public static String loading(String name) {
494    public static String quit(Scanner Sc, String name) {
498    public static void main(String[] args) {
499        start();
500    }
501 }
```

This screenshot helps you to understand flow of the methods.



This is the main flow chart of methods.

“quit()” method is reachable from every method .

1-)Program starts with “start()” method it returns “get_age()” and get_age method takes the user’s name from “get_name()” method.Then get_age method returns “get_position()” method.

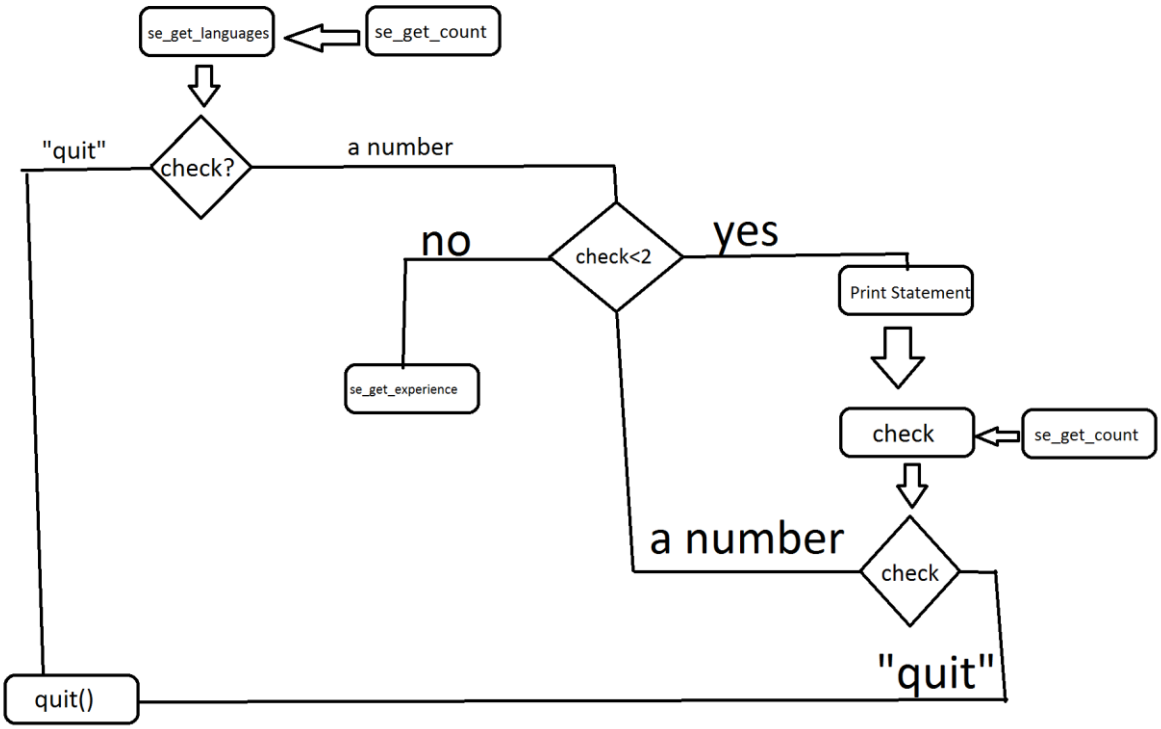
2-)get_position method divided into three sections based on user’s input.

3-)Method’s flows as rings of a chain.

4-)”se_get_count()” and “acc_get_count()”methods reachable for Software Engineer section and Accountant sections, which were explained in code in terms of comment statements. But for now we can denote that these methods check wheter the user satisfies at least two of three conditions.

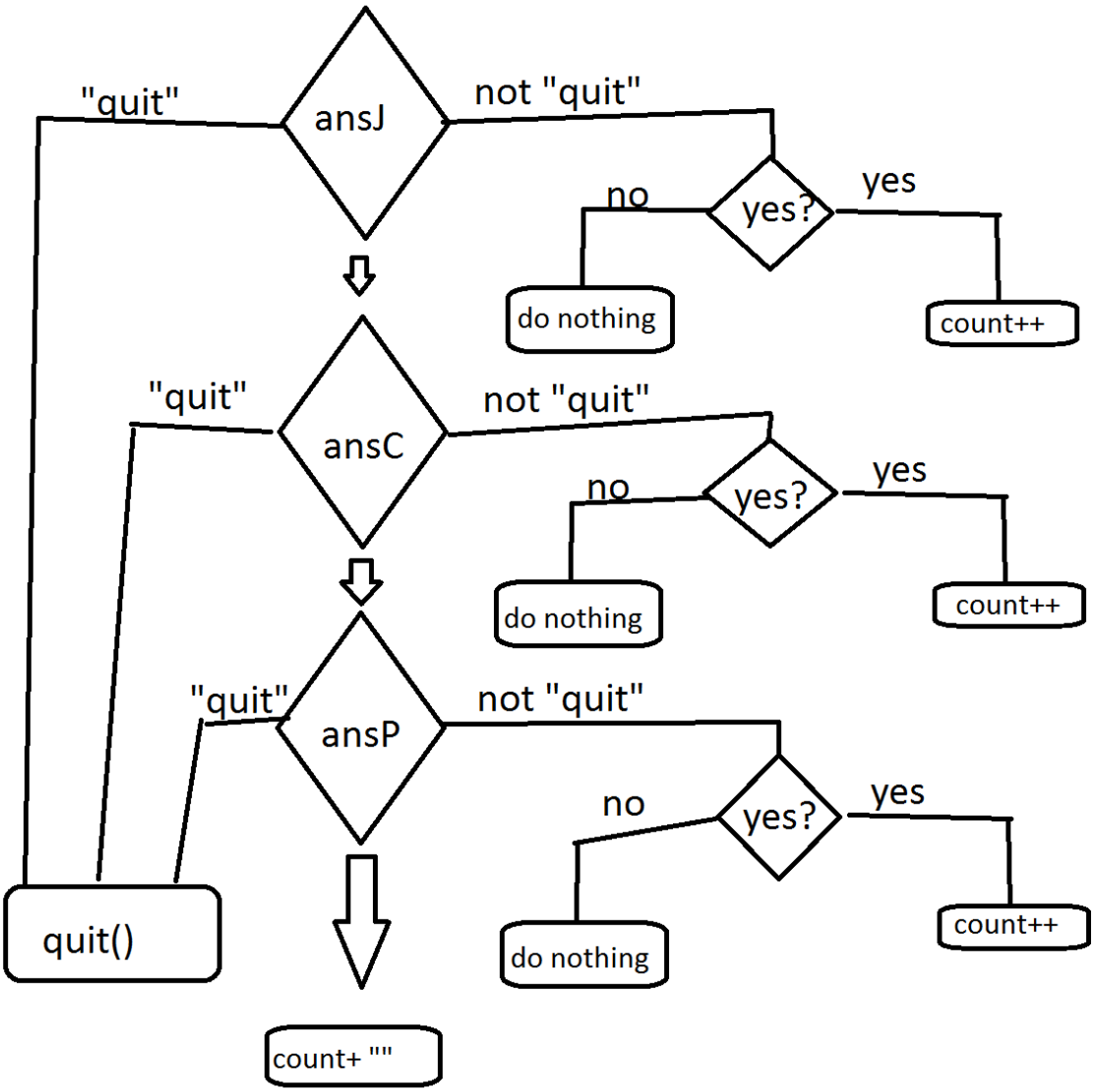
5-)All three sections reach to the “get_gender()” method.

```java
public static String se_get_languages(Scanner sc, String name) {
    /*
     * To reduce the complexity in this method I created "se_get_count" method.
     * "se_get_count" method could be considered as submethod of "se_get_languages".
     * Therefore, I started to write it 8 white-space inner.
     * "se_get-languages" method takes all information from "se_get_count" method.
     * Therefore, in order to understand this method, they need to be considered together.
     * This method could be little complicated but
     * I'm going to explain it more detailed next to the statements.
     */
    System.out.println("We want our employees to be experienced in at least two programming languages among Java, C++, and Python.");
    String check =se_get_count(sc); //First, program runs "se_get_count" method and assigns it value to even if it is a number or "quit".

    if(check.equalsIgnoreCase("quit"))//Then checks whether it is quit or not.
        return quit(sc,name);

    //If program comes this line we know that check is not "quit" which means user did not write "quit" at first try.
    while(Integer.parseInt(check)<2){ //Because of check's type is String, we need to "parse" it to integer so that we can compare its value
        System.out.println("In order to get accepted to this position, you have to be experienced in at least two programming languages, if so, give it another try.");
        check=se_get_count(sc);//Due to the chance of misspelling by user, we want her/him to answer questions again.

        if(check.equalsIgnoreCase("quit"))//This statement checks user's input whether it is "quit" or not, for second and later tries.
            return quit(sc,name);

    }
    //If program comes this line we know that user did not write "quit" and s/he knows more than 2 programming languages.
    return se_get_experience(sc,name);
}
```



As an example, code and flow chart of “se_get_languages()” method.

```java
public static String se_get_count(Scanner sc) {
    /*
     * This method counts the number of languages the user know.
     */
    int count =0;//Counts the number of languages known by user.
    System.out.println("Are you experienced in Java?");
    String ansJ = sc.nextLine();
    if(ansJ.equalsIgnoreCase("quit"))//"quit" method is still available at this point, because the type of method is String.
        return "quit";
    if(ansJ.equalsIgnoreCase("yes"))
        count++;
    System.out.println("Are you experienced in C++?");
    String ansC = sc.nextLine();
    if(ansC.equalsIgnoreCase("quit"))
        return "quit";
    if(ansC.equalsIgnoreCase("yes"))
        count++;
    System.out.println("Are you experienced in Python?");
    String ansP = sc.nextLine();
    if(ansP.equalsIgnoreCase("quit"))
        return "quit";
    if(ansP.equalsIgnoreCase("yes"))
        count++;
    return count+""; //I added "" at end of the statement because count is integer but we need to return a String.
}
```



As an example, code and flow chart of “se_get_count()” method.

My code's some methods really similar to each other. For example:

- se_get_university
- se_get_field
- acc_get degree
- acc_get_excel
- acc_get_dLicence
- academy_get_english
- academy_get_teach

methods really similar to each other.

On the other hand,

- se_get_languages
- acc_get_people
- academy_get_papers

are similar too. Therefore, "se_get_count" and "acc_get_count" methods are almost the same. But academy_get_papers method does not have count submethod, because in that method we only take the number of papers candidate has.

3-)Implementation

```
import java.util.Scanner;


public class EG2016400099{
	public static void start() {
		/*
		 * This method is created to give main method more proper code.
		 * I could have write this statements into the main but in order to maintain it simple
		 * this method is used.
		 * It invokes "get_age" method which is the starting of flow of methods.
		 * It is going to be explained later more detailed.
		 */
		System.out.println("Welcome to MARS Interviewing Systems");
		Scanner sc = new Scanner(System.in);
		System.out.println(get_age(sc));
	}
	//-----IN COMMON-----
	/*
	 * This part is in common for all positions.
```

```
	 * Gets name and age of applicant.Then asks her/him which position s/he wants.
	 * I have connected methods to each other.
	 * Program starts with "get_age" methods and it invokes "get_name" method while calling to user with her/his name.
	 * Then get_age method returns "get_position" method.
	 * Furthermore, get_position method returns other methods based on input received from user.
	 * Therefore, it implys program to related method block such as Accountant.
	 * In addition, from "get_position" on, "quit" method is available.
	 * Which allows user to end interview whenever s/he wants.
	 * And "name" parameter allows us to carry user's name to other methods.
	 * Each methods take name as parameter and returns it in order to transfer it to the next method.
	 * More detailed comments are exist in each method block.
	 */
	public static String get_name(Scanner sc) {
		/*
		 * This method takes a token in type String and returns it.
		 * The only function of this method is this.
		 * And it is invoked in "get_age".
		 */
		System.out.println("Can we learn your name?");
		String name = sc.nextLine();
		return name;
	}
	public static String get_age(Scanner sc) {
		/*
		 * This method is the starting point of my code.
		 * It starts "get_name" method in line 45. as a part of print statement.So we take user's name.
		 * It also takes an integer and assign it to "age" variable.
		 * It checks that this input is bigger than 18 or not negative.
```

```
		 * If everything is valid it invokes get_position method.
		 */
		String name = get_name(sc); //This statement allows us to pass name as parameter to the next methods.Otherwise, I could invoke "get_name" in the print statement.
		System.out.println("Dear "+ name + ", could you please share your age with us?");
		int age = Integer.parseInt(sc.nextLine());//Because of "nextLine()" method creates problem with "nextInt()" method, I used "parse" which is allowed personally to me by out Teaching Assistant Uras Mutlu.
		while(age<19 ){
			if(age>0) {
				System.out.println("Sorry "+name+".You have to be older than 18 in order to apply this application.Please write a valid value.");
				age = Integer.parseInt(sc.nextLine());
			}
			else if(age<=0) {
				System.out.println("The value you submitted is not valid. Please write a positive integer.Please write a valid value.");
				age = Integer.parseInt(sc.nextLine());
			}
		}
		return get_position(sc,name);
	}
	public static String get_position(Scanner sc, String name) {
		/*
		 * In this method program determines the way which is going to be followed.
		 * This step is kind of distributing point that determines which questions program is going to ask to the user.
		 * I separated question blocks for each position with comment statements so it is more readable.
		 * while block is used to ensure user wrote a valid expression.
		 * In addition, I used "equalsIgnoreCase" method so, program does not expect a certain expression, which means it is user-friendly.
		 * Rest of the my code "equalsIgnoreCase" method is used for same reason so, I will not explain it again.
```

```
		 * If expression is not valid program is going to keep saying to user that s/he has to write a valid expression.

		 * print statements are well described in order to help user.

		 */

		System.out.println("Dear "+name+".From now on, you can quit the interview whenever you want by writing \"Quit\".");

		System.out.println("For which position are you applying ?");

		String position = sc.nextLine();

		//-----QUIT-----

		//To emphasize the quit statement and make it more readable,

		//I left an empty line before and after the quit statement along the whole code.


		if(position.equalsIgnoreCase("quit"))

			return quit(sc,name);


		while(!position.equalsIgnoreCase("software engineer") && !position.equalsIgnoreCase("accountant") && !position.equalsIgnoreCase("academic")) {

			System.out.println("Sorry "+name+".The positions our company is looking for are \"Software Engineer\" , \"Accountant\" or \"Academic\". \nIf you sure that you are applying for one of them, give it another try now and please make sure that you have passed your input correctly.");

			position = sc.nextLine();


			if(position.equalsIgnoreCase("quit"))

				return quit(sc,name);


		}

		if(position.equalsIgnoreCase("software engineer")) {

			System.out.println("Great! This is the exact position we are looking for!");

			return se_get_university(sc,name);

		}

			else if(position.equalsIgnoreCase("accountant")) {

				System.out.println("Great! This is the exact position we are looking for!");
```

```
                return acc_get_degree(sc,name);
            }
                else {
                    System.out.println("Great! This is the exact position we are looking for!");
                    return academy_get_english(sc,name);
                }
        }
        //-----SOFTWARE ENGINEER-----
        /*
         * "se_" part of the methods' name represents "sOFTWARE eNGINEER".
         * In this part, I have connected methods too but more linear way.
         * Methods return one another respectively just like a chain ring.
         * They are following each other in order they were coded.
         * And "name" parameter allows us to carry user's name to other methods.
         * Each methods take "name" as parameter and returns it in order to transfer it to the next method.
         * "se_get_count" method is an exception but it is explained in its block.
         * More detailed comments are exist in each method block.
         */
        public static String se_get_university(Scanner sc, String name) {
            /*
             * In this method program questions the existence of university degree.
             * Again, while statements are used to get valid expression from user.
             */
            System.out.println("Do you have a university degree?");
            String degree = sc.nextLine();

            if(degree.equalsIgnoreCase("quit"))
                return quit(sc,name);

            while(!degree.equalsIgnoreCase("yes")) {
```

```
                System.out.println("Sorry "+name+".We are looking for a person having university degree.\nIf you have, please write \"Yes\"");
                degree = sc.nextLine();

                if(degree.equalsIgnoreCase("quit"))
                        return quit(sc,name);

        }
        return se_get_field(sc,name);
    }
    public static String se_get_field(Scanner sc, String name) {
        //Very similar to "se_get_university" method.
        System.out.println("In which field?");
        String field = sc.nextLine();

        if(field.equalsIgnoreCase("quit"))
                return quit(sc,name);

        while(!field.equalsIgnoreCase("software engineering") && !field.equalsIgnoreCase("computer engineering") && !field.equalsIgnoreCase("computer science")) {
                System.out.println("Sorry "+name+".We are looking for a person having a univsersity degree in:\n1-)\"Software Engineering\"\n2-)\"Computer Engineering\"\n3-)\"Computer Science\"");
                System.out.println("If you sure that you have one of them, give it another try now and please make sure that you have passed your input correctly.");
                field = sc.nextLine();

                if(field.equalsIgnoreCase("quit"))
                        return quit(sc,name);

        }
        System.out.println("Very nice!");
        return se_get_languages(sc,name);
```

```
	}
	public static String se_get_languages(Scanner sc, String name) {
		/*
		 * To reduce the complexity in this method I created "se_get_count" method.
		 * "se_get_count" method could be considered as submethod of "se_get_languages".
		 * Therefore, I started to write it 8 white-space inner.
		 * "se_get-languages" method takes all information from "se_get_count" method.
		 * Therefore, in order to understand this method, they need to be considered together.
		 * This method could be little complicated but
		 * I'm going to explain it more detailed next to the statements.
		 */
		System.out.println("We want our employees to be experienced in at least two programming languages among Java, C++, and Python.");
		String check =se_get_count(sc); //First, program runs "se_get_count" method and assigns it value to even if it is a number or "quit".

		if(check.equalsIgnoreCase("quit"))//Then checks whether it is quit or not.
			return quit(sc,name);

		//If program comes this line we know that check is not "quit" which means user did not write "quit" at first try.
		while(Integer.parseInt(check)<2){ //Because of check's type is String, we need to "parse" it to integer so that we can compare its value
			System.out.println("In order to get accepted to this position, you have to be experienced in at least two programming languages, if so, give it another try.");
			check=se_get_count(sc);//Due to the chance of misspelling by user, we want her/him to answer questions again.

			if(check.equalsIgnoreCase("quit"))//This statement checks user's input whether it is "quit" or not, for second and later tries.
				return quit(sc,name);
```

```
		}
		//If program comes this line we know that user did not write "quit" and s/he knows more than 2 programming languages.
		return se_get_experience(sc,name);
	}
		public static String se_get_count(Scanner sc) {
			/*
			 * This method counts the number of languages the user know.
			 */
			int count =0;//Counts the number of languages known by user.
			System.out.println("Are you experienced in Java?");
			String ansJ = sc.nextLine();
			if(ansJ.equalsIgnoreCase("quit"))//"quit" method is still available at this point, because the type of method is String.
				return "quit";
			if(ansJ.equalsIgnoreCase("yes"))
				count++;
			System.out.println("Are you experienced in C++?");
			String ansC = sc.nextLine();
			if(ansC.equalsIgnoreCase("quit"))
				return "quit";
			if(ansC.equalsIgnoreCase("yes"))
				count++;
			System.out.println("Are you experienced in Python?");
			String ansP = sc.nextLine();
			if(ansP.equalsIgnoreCase("quit"))
				return "quit";
			if(ansP.equalsIgnoreCase("yes"))
				count++;
			return count+""; //I added "" at end of the statement because count is integer but we need to return a String.
		}
```

```
	public static String se_get_experience(Scanner sc, String name) {
		/*
		 * In this method at least one of the answers must be "yes".
		 * If answer of first question is yes, program will immediately jump to the return line.
		 * Because condition is satisfied.
		 * If first answer is not "yes", it is going to ask second question.
		 * If second answer is "yes", program will jump to return line without running while statement.
		 * If second answer is not "yes", while statement will force user to write "yes".
		 */
		System.out.println("Nice.Have you ever worked as a software engineer at least three years before?");
		String work = sc.nextLine();

		if(work.equalsIgnoreCase("quit"))
			return quit(sc,name);

		if(!work.equalsIgnoreCase("yes")) {	//This statement causes method to immediately return "get_gender" method, in case s/he has the experience.
			//Because if user have 3 years experience, we do not need to question him whether s/he has graduate degree or not.
			System.out.println("Do you have a graduate degree in software engineering?");
			String graduate = sc.nextLine();

			if(graduate.equalsIgnoreCase("quit"))
				return quit(sc,name);

			while(!work.equalsIgnoreCase("yes") && !graduate.equalsIgnoreCase("yes")) {
				System.out.println("Sorry "+name+".You have to either have more than three years of experience as a software engineer or \ngraduate degree in software engineering.");
```

```
                                System.out.println("Please write \"Yes\" if you satisfy the
condition.");
                                System.out.println("Have you ever worked as a software engineer
before?");
                                work = sc.nextLine();

                                if(work.equalsIgnoreCase("quit"))
                                        return quit(sc,name);

                                System.out.println("Do you have a graduate degree in software
engineering?");
                                graduate = sc.nextLine();

                                if(graduate.equalsIgnoreCase("quit"))
                                        return quit(sc,name);

                        }
                }
                return get_gender(sc,name);// End of the specialized part of interview for "Software
Engineer".
        }
        //-----ACCOUNTANT-----
        /*
         * "acc_" part of the methods' name represents "accOUNTANT".
         * This block really similar to Software Engineer block.
         * I used chain-ring model again.
         * Therefore, comments in that block are not going to be that much.
         * In order to understand this block better, you should check whether you understood
previous block well or not.
         */
        public static String acc_get_degree(Scanner sc, String name) {
                //Very similar to "se_get_university" method.
```

```
		System.out.println("Do you have accountant degree?");//Questions if user has accountant degree or not.
		String degree = sc.nextLine();

		if(degree.equalsIgnoreCase("quit"))
			return quit(sc,name);

		while(!degree.equalsIgnoreCase("yes")) { //In case a misspelling occurs, program wants user to rewrite.
			System.out.println("Sorry "+name+"."+"In order to get accepted to this position, you have to have accountant degree.\nIf you do, please make sure that you write \"Yes\"");
			degree = sc.nextLine();

			if(degree.equalsIgnoreCase("quit"))
				return quit(sc,name);

		}
		return acc_get_excel(sc,name);
	}
	public static String acc_get_excel(Scanner sc, String name) {
		//Very similar to acc_get_degree method.
		System.out.println("Do you know Excel well?");
		String excel = sc.nextLine();

		if(excel.equalsIgnoreCase("quit"))
			return quit(sc,name);

		while(!excel.equalsIgnoreCase("yes")) {
			System.out.println("Sorry "+name+"."+"In order to get accepted to this position, you have to know how to use Excel.\nIf you do, please make sure that you write \"Yes\"");
			excel = sc.nextLine();
```

```
                if(excel.equalsIgnoreCase("quit"))
                        return quit(sc,name);


        }
        return acc_get_english(sc,name);
    }
    public static String acc_get_english(Scanner sc, String name) {
        //Very similar to "se_get_experience" method.
        System.out.println("Can you speak English fluently?");
        String fluent = sc.nextLine();

        if(fluent.equalsIgnoreCase("quit"))
                return quit(sc,name);

        if(!fluent.equalsIgnoreCase("yes")) {
                System.out.println("Do you have a friend who can translate texts from
English to your native language?");
                String friend = sc.nextLine();

                if(friend.equalsIgnoreCase("quit"))
                        return quit(sc,name);

                while(!fluent.equalsIgnoreCase("yes") && !friend.equalsIgnoreCase("yes")) {
                        System.out.println("Sorry "+name+"."+"You have to either speak
fluent English or have a friend who can translate for you.");
                        System.out.println("Please write \"Yes\" if you satisfy the
condition.");
                        System.out.println("Can you speak English fluently?");
                        fluent= sc.nextLine();

                        if(fluent.equalsIgnoreCase("quit"))
                                return quit(sc,name);
```

```
                    System.out.println("Do you have a friend who can translate texts from English to your native language?");
                    friend = sc.nextLine();

                    if(fluent.equalsIgnoreCase("quit"))
                        return quit(sc,name);

                }
            }
            return acc_get_people(sc,name);
        }
        public static String acc_get_people(Scanner sc, String name) {
            //Very similar to "se_get_languages" method.
            System.out.println("We want our new employees to be known among our current experienced employees \nsuch as Pınar Yolum Birbil, Uras Mutlu, and Emre Girgin");
            String check =acc_get_count(sc);

            if(check.equalsIgnoreCase("quit"))
                return quit(sc,name);

            while(Integer.parseInt(check)<2){
                System.out.println("In order to get accepted to this position, you have to be known among our current experienced eployees, if so, give it another try.");
                check=acc_get_count(sc);

                if(check.equalsIgnoreCase("quit"))
                    return quit(sc,name);
            }
            return "Great" + acc_get_dLicense(sc,name);
        }
            public static String acc_get_count(Scanner sc) {
```

```
		//Very similar to "se_get_count" method.
		int count =0;
		System.out.println("Are you known by Pınar Yolum Birbil");
		String ansP = sc.nextLine();
		if(ansP.equalsIgnoreCase("quit"))
			return "quit";
		if(ansP.equalsIgnoreCase("yes"))
			count++;
		System.out.println("Are you known by Uras Mutlu");
		String ansU = sc.nextLine();
		if(ansU.equalsIgnoreCase("quit"))
			return "quit";
		if(ansU.equalsIgnoreCase("yes"))
			count++;
		System.out.println("Are you known by Emre Girgin");
		String ansE = sc.nextLine();
		if(ansE.equalsIgnoreCase("quit"))
			return "quit";
		if(ansE.equalsIgnoreCase("yes"))
			count++;
		return count+"";
	}
public static String acc_get_dLicense(Scanner sc, String name) {
	//Very similar to "acc_get_degree" method.
	System.out.println("Do you have driving license?");
	String dLicense = sc.nextLine();

	if(dLicense.equalsIgnoreCase("quit"))
		return quit(sc,name);

	while(!dLicense.equalsIgnoreCase("yes")) {
```

```
                System.out.println("Sorry "+name+"."+"In order to get accepted to this position, you have to have driving licese.\nIf you do, please make sure that you write \"Yes\"");
                dLicense = sc.nextLine();

                if(dLicense.equalsIgnoreCase("quit"))
                        return quit(sc,name);

        }
        return get_gender(sc,name);// End of the specialized part of interview for "Accountant".
    }
    //-----ACADEMIC-----
    //-----ACADEMIC-----
    /*
     * "academy_" part of the methods' name represents "ACADEMIC".
     * This block really similar to Software Engineer and Accountant block.
     * I used chain-ring model again.
     * Therefore, comments in that block are not going to be that much.
     * In order to understand this block better, you should check whether you understood previous block well or not.
     */
    public static String academy_get_english(Scanner sc, String name) {
        //Very similar to "se_get_university" method.
        System.out.println("Can you speak English?");
        String academy_english = sc.nextLine();

        if(academy_english.equalsIgnoreCase("quit"))
                return quit(sc,name);

        while(!academy_english.equalsIgnoreCase("yes")) {
                System.out.println("Sorry "+name+"."+"In order to get accepted to this position, you have to speak English.\nIf you do, please make sure that you write \"Yes\"");
```

```
                academy_english = sc.nextLine();


                if(academy_english.equalsIgnoreCase("quit"))
                        return quit(sc,name);


        }
        return academy_get_papers(sc,name);
    }
    public static String academy_get_papers(Scanner sc, String name) {
        /*
         * It is similar to "se_get_languages" method but with little difference.
         * In this method we do not have a count method but we still need to parse.
         */
        System.out.println("Good.How many published papers do you have?(Please write it as figures)");
        String check=sc.nextLine();


        if(check.equalsIgnoreCase("quit"))//Checks whether user write "quit" or not.
                return quit(sc,name);
        //Then...
        while(Integer.parseInt(check)<3) {//Uses "parse" to compare with test condition.
                System.out.println("Sorry "+name+".In order to get accepted to this position, you have to have at least three published papers.\nIf you do, please make sure that you write an integer bigger than or equal to three as figure.");
                check=sc.nextLine();


                if(check.equalsIgnoreCase("quit"))
                        return quit(sc,name);
        }
        return academy_get_teach(sc,name);
    }
    public static String academy_get_teach(Scanner sc, String name) {
```

```
		//Very similar to "academy_get_english" method.
		System.out.println("Very nice!Do you love teaching?");
		String teaching = sc.nextLine();

		if(teaching.equalsIgnoreCase("quit"))
			return quit(sc,name);

		while(!teaching.equalsIgnoreCase("yes")) {
			System.out.println("Sorry "+name+"."+"In order to get accepted to this position, you'd better love teaching.\nIf you have, please make sure that you write \"Yes\"");
			teaching = sc.nextLine();

			if(teaching.equalsIgnoreCase("quit"))
				return quit(sc,name);

		}
		return get_gender(sc,name);// End of the specialized part of interview for "Academic".
	}
	//-----GENERAL-----
	//-----GENERAL-----
	/*
	 * These three methods are in common for all positions.
	 * This block contains three blocks but only "get_gender" method worth to look in.
	 * More detailed comments are exist in methods.
	 */
	public static String get_gender(Scanner sc, String name) {
		System.out.println("Nice.Are you male?");
		String gender = sc.nextLine();

		if(gender.equalsIgnoreCase("quit"))
			return quit(sc,name);
```

```
            if(gender.equalsIgnoreCase("yes")) {// If user is male, military service block runs.
                System.out.println("Have you completed your military service?");
                String military = sc.nextLine();

                if(military.equalsIgnoreCase("quit"))
                    return quit(sc,name);

                while(!military.equalsIgnoreCase("yes")) {//Asks user to rewrite the answer again and again unless he writes "quit".
                    System.out.println("Sorry "+name+"."+"We are looking for someone who have completed his military service.\nIf you have, please make sure that you write \"Yes\"");
                    military = sc.nextLine();

                    if(military.equalsIgnoreCase("quit"))
                        return quit(sc,name);

                }
            }
            return loading(name);
        }
        public static String loading(String name) {
            //This method is only for fun and result.
            System.out.println("Your application is being processed. Please wait.");
            for(int i=1;i<=5;i++) {
                System.out.println("%"+(i*20));
            }
            return "Congratulations " +name+ "!You got the job!";
        }
        public static String quit(Scanner Sc, String name) {
            //This method allow user to quit the interview.
            return "Dear "+name+".Thank you for your interest.The interview is done.";
```

```
        }

        public static void main(String[] args) {

                start();

        }

}
```

4-)Outputs of the program

```
Welcome to MARS Interviewing Systems
Can we learn your name?
Emre Girgin
Dear Emre Girgin, could you please share your age with us?
19
Dear Emre Girgin.From now on, you can quit the interview whenever you want by writing "Quit".
For which position are you applying ?
Software Engineer
Great! This is the exact position we are looking for!
Do you have a university degree?
Yes
In which field?
Computer Engineering
Very nice!
We want our employees to be experienced in at least two programming languages among Java, C++, and Python.
Are you experienced in Java?
Yes
Are you experienced in C++?
Yes
Are you experienced in Python?
No
Nice.Have you ever worked as a software engineer at least three years before?
no
Do you have a graduate degree in software engineering?
Yes
Nice.Are you male?
yes
Have you completed your military service?
yes
Your application is being processed. Please wait.
%20
%40
%60
%80
%100
Congratulations Emre Girgin!You got the job!
```

```
Welcome to MARS Interviewing Systems
Can we learn your name?
robot_42
Dear robot_42, could you please share your age with us?
55
Dear robot_42.From now on, you can quit the interview whenever you want by writing "Quit".
For which position are you applying ?
Cleaning
Sorry robot_42.The positions our company is looking for are "Software Engineer" , "Accountant" or "Academic".
If you sure that you are applying for one of them, give it another try now and please make sure that you have passed your input correctly.
Quit
Dear robot_42.Thank you for your interest.The interview is done.
```

```
Welcome to MARS Interviewing Systems
Can we learn your name?
Matt
Dear Matt, could you please share your age with us?
25
Dear Matt.From now on, you can quit the interview whenever you want by writing "Quit".
For which position are you applying ?
Software Engineer
Great! This is the exact position we are looking for!
Do you have a university degree?
yes
In which field?
Programming
Sorry Matt.We are looking for a person having a univsersity degree in:
1-)"Software Engineering"
2-)"Computer Engineering"
3-)"Computer Science"
If you sure that you have one of them, give it another try now and please make sure that you have passed your input correctly.
Computer Science
Very nice!
We want our employees to be experienced in at least two programming languages among Java, C++, and Python.
Are you experienced in Java?
no
Are you experienced in C++?
no
Are you experienced in Python?
no
In order to get accepted to this position, you have to be experienced in at least two programming languages, if so, give it another try.
Are you experienced in Java?
quit
Dear Matt.Thank you for your interest.The interview is done.
```

```
Welcome to MARS Interviewing Systems
Can we learn your name?
Roberto
Dear Roberto, could you please share your age with us?
36
Dear Roberto.From now on, you can quit the interview whenever you want by writing "Quit".
For which position are you applying ?
Accountant
Great! This is the exact position we are looking for!
Do you have accountant degree?
yes
Do you know Excel well?
yes
Can you speak English fluently?
no
Do you have a friend who can translate texts from English to your native language?
yes
We want our new employees to be known among our current experienced employees
such as Pınar Yolum Birbil, Uras Mutlu, and Emre Girgin
Are you known by Pınar Yolum Birbil
yes
Are you known by Uras Mutlu
yes
Are you known by Emre Girgin
no
Do you have driving license?
no
Sorry Roberto.In order to get accepted to this position, you have to have driving licese.
If you do, please make sure that you write "Yes"
quit
GreatDear Roberto.Thank you for your interest.The interview is done.
```

5-)Conclusion

My program works as it is wanted. In addition, it keeps asking when inputs do not satisfy the expectation of program in case a misspeling occurs. Also, user can leave the program whevever s/he wants due to lack of quality.