MiniProjectLevel1

October 13, 2020

1 Mini Project : Help Bot

2 Level 1

2.1 Anis Faqiehah binti Zainudin

2.2 11 September 2020

2.3 Version 1

2.4 Summary of the Question

Write a help bot procedural program that can give help in a realistic way answering text questions about a specific subject (eg cooking / gardening problems, how to play Minecraft, etc)

2.5 The literate program development

2.5.1 helloMessage

What it does This method prints message to screen.

Implementation (how it works) This program introduces it self by using System.out.println("The message goes here!"); The message goes inside the brackets and inside the quotes.

[1]:
```
public static void helloMessage ()
{
     System.out.println("Hello!");
     System.out.println("I'm Mr. Meeseeks. Look at me!");
     return;
} // END helloMessage
```

Testing

[2]:
```
helloMessage();
```

```
Hello!
I'm Mr. Meeseeks. Look at me!
```

2.5.2 askQuestion1

What it does This method allows us to type in any question we want to ask regarding the game Valorant.

Implementation (how it works) The println commands just print messages. There is a command stated as

question1 = scanner.nextLine(); asks the user to type something waiting until they do. Whatever they type is stored as question1.

[8]:
```
public static void askQuestion1 ()
{
    String question1;
    Scanner scanner = new Scanner(System.in); //Make scanner available

    System.out.println("What do you want to know about this game?");
    question1 = scanner.nextLine();

    System.out.println("I'm sorry, I'm not sure about that.");

    return;
} // END askQuestion1
```

Testing

[9]:
```
askQuestion1();
```

```
What do you want to know about this game?
What is CT?
I'm sorry, I'm not sure about that.
```

2.5.3 Running the program

Run the following call to simulate running the complete program.

[11]:
```
helloMessage();
askQuestion1();
```

```
Hello!
I'm Mr. Meeseeks. Look at me!
What do you want to know about this game?
Who is Jett?
I'm sorry, I'm not sure about that.
```

2.6 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[ ]:
```
// ANIS FAQIEHAH BINTI ZAINUDIN
// 11 SEPTEMBER 2020
// VERSION 1
// A HELP BOT THAT ANSWERS EVERYTHING

import java.util.Scanner; // Needed to make Scanner available

class helpbot // change the name to helpbot
{
    public static void main (String [] a)
    {
        helloMessage();
        askQuestion1();
        System.exit(0);
    }

    public static void helloMessage () // Intro of helpbot
    {
     System.out.println("Hello!"); // Greetings
     System.out.println("I'm Mr. Meeseeks. Look at me!"); //greetings 2
     return;
    } // END helloMessage

    public static void askQuestion1 () // Question to input by user
    {
    String question1;
    Scanner scanner = new Scanner(System.in); //Make scanner available

    System.out.println("What do you want to know about this game?"); // Helpbot␣
 ↪asks first
    question1 = scanner.nextLine();

    System.out.println("I'm sorry, I'm not sure about that."); // Answers to␣
 ↪any question

    return;
    } // END askQuestion1

}
```

END OF LITERATE DOCUMENT

MINIPROJECT_level2

October 31, 2020

1 Mini Project Number 4 : Help Bot

2 Level 2

3 Anis Faqiehah binti Zainudin

3.1 27/10/2020

3.2 Version 1

3.3 Summary of the Question

Write a help bot procedural program that can give help in a realistic way answering text questions about a specific subject (eg cooking / gardening problems, how to play Minecraft, etc)

As above, but the helpbot now answers in a different way depending on the question asked. A separate 'choice' method is passed the person's question as argument and prints the appropriate answer. Things the helpbot says are now stored in local String variables with separate statements concatenated together.

3.4 The literate program development

3.4.1 inputString

What it does This method takes a String question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) It sets up a local scanner to the method, linked to the keyboard. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole String up to ENTER as the response). Whatever typed is stored in a String, and returned as the result of the method.

[9]:
```
// Method that includes
public static String inputString(String message)
    {
       Scanner scanner = new Scanner(System.in);
       String answer;

       System.out.println(message);
       answer = scanner.nextLine();
```

```
    return answer;
  } // END inputString
```

Testing

[10]:
```
String agent = inputString("What do you want to know about?");
```

```
What do you want to know about?
Anis
```

3.4.2 print

What it does This method takes a String argument which is the message to print. It prints that message and returns. The cursor is left on the next line for anything subsequently printed.

Implementation (how it works) The string passed when the method is called is stored in the variable called message. This is then passed directly on to System.out.println which does the actual printing. It then returns.

[11]:
```
public static String print(String message) // Print out messages
{
   System.out.println(message);

   return message;
} // END print
```

Testing

[12]:
```
print(agent + "Brimstone");
```

```
AnisBrimstone
```

[12]:
```
AnisBrimstone
```

3.4.3 Running the program

Run the following call to simulate running the complete program.

[13]:
```
/* ASKS ABOUT PREFERRED AGENT'S CLASS */
String ans = inputString("What class of agent do you prefer to play? Duelist/
 ↪Controller/Sentinels/Initiator");
String agent = "The best agent for you is: ";

if (ans.contains("Controller"))
{
    print(agent + "Brimstone");
}
else if (ans.contains("Sentinels"))
```

```
{
    print(agent + "Sage");
}
else if (ans.contains("Initiator"))
{
    print(agent + "Sova");
}
else if (ans.contains("Duelist"))
{
    print(agent + "Reyna");
}
else
{
    print("I'm not sure about that. Try again?(Duelist/Controller/Sentinels/
 ↪Initiator)");
}
```

```
What class of agent do you prefer to play?
Duelist/Controller/Sentinels/Initiator
Duelist
The best agent for you is: Reyna
```

3.5 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[3]:
```
// ANIS FAQIEHAH BINTI ZAINUDIN
// 27 OCTOBER 2020
// VERSION 1
// A HELP BOT THAT ANSWERS EVERYTHING

import java.util.*; //import all

class helpbot // named to helpbot
{

    // Ask a question about what and try and say something relevant to the␣
 ↪answer.
    public static void main (String[] param)
    {
        String ans = inputString("What class of agent do you prefer to play?␣
 ↪Duelist/Controller/Sentinels/Initiator");
        String agent = "The best agent for you is: ";

       if (ans.contains("Controller"))
       {
```

```
            print(agent + "Brimstone");
        }
        else if (ans.contains("Sentinels"))
        {
            print(agent + "Sage");
        }
        else if (ans.contains("Initiator"))
        {
            print(agent + "Sova");
        }
        else if (ans.contains("Duelist"))
        {
            print(agent + "Reyna");
        }
        else // If none of the choices above were stated
        {
            print("I'm not sure about that. Try again?(Duelist/Controller/
↪Sentinels/Initiator)");
        }

        System.exit(0);

    } // END main



    /* ********************************
    Ask for a string with given message
    Return the string typed in by the user
    */
    public static String inputString(String message)
    {
        Scanner scanner = new Scanner(System.in);
        String answer;

        System.out.println(message);
        answer = scanner.nextLine();

        return answer;
    } // END inputString

    public static String print(String message) // Print out messages
    {
        System.out.println(message);

        return message;
    } // END print
```

```
} // END helpbot
```

END OF LITERATE DOCUMENT

MiniProjectLevel3

December 8, 2020

1 Mini Project(Help bot)

2 Level 3

2.1 Anis Faqiehah binti Zainudin

2.2 1 November 2020

2.3 Version 1

2.4 Summary of the Question

As before, but now when the person asks a question it looks if the person's answer contains a specific 'trigger' word (ie a word the program knows a response for). That trigger word and a response to it is stored as a record. The record includes a field for the trigger and a field for the response (eg if "greenfly" is in a question that would lead an answer "Encourage ladybirds." then those two strings are stored in, and accessed from, different fields of the record). The record is passed as a single argument to the method(s) that look for the response and gives it.

The program also uses a loop so that when the question is answered it invites the user to ask another until someone types "Goodbye" at which point the program ends.

2.5 The literate program development

[9]:
```
class Agent
{
    String origin;
    boolean heal;
    int ultimate;
}
```

2.5.1 InputString

What it does This method takes a String question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) It sets up a local scanner to the method, linked to the keyboard. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole String up to ENTER as the response). Whatever typed is stored in a String, and returned as the result of the method.

[1]:
```
public static String inputString (String message)
{
        String answer;
        Scanner scanner = new Scanner(System.in);

        System.out.println(message);
        answer = scanner.nextLine();

        return answer;
} // END inputString
```

Testing

[ ]:
```
String square = inputString("What size square of numbers do you want?");
```

2.5.2 InputInt

What it does This method takes an integer question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) The first variable creates a scanner to allow keyboard input. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole integer up to ENTER as the response). Whatever typed is stored in a integer, and returned as the result of the method.

[2]:
```
// input an integer
//
public static int inputInt (String message)
{
        String answer = inputString(message);

        return Integer.parseInt(answer);
}
```

Testing

[ ]:
```
int number = inputInt("How old are you?");
```

2.5.3 gameDetails

What it does This method creates a record of each Agent. The following is asked for the user to input the answers.

- How many questions do you have?
- What do you need to know about the agents?

In each case immediately after asking it waits for the answer and then echos it back in a common format.

Implementation (how it works) Using records and for loops. Inside the for loops, there's if statements.

[5]:
```
public static void gameDetails()
    {
        Agent Breach = createAnswer("Sweden", false, 7);
        Agent Brimstone = createAnswer("United States", false, 6);
        Agent Cypher = createAnswer("Morocco", false, 7);
        Agent Omen = createAnswer("Unknown", false, 7);
        Agent Killjoy = createAnswer("Germany", false, 7);
        Agent Raze = createAnswer("Brazil", false, 6);
        Agent Sova = createAnswer("Russia", false, 7);
        Agent Jett = createAnswer("South Korea", false, 6);
        Agent Sage = createAnswer("China", true, 7);
        Agent Phoenix = createAnswer("United Kingdom", true, 6);
        Agent Reyna = createAnswer("Mexico", true, 6);
        Agent Viper = createAnswer("United States", false, 7);
        Agent Skye = createAnswer("Australia", true, 6);

        final int n = inputInt("How many questions do you have?");

        for(int i = 1; i<=n; i++)
        {
            String q2 = inputString("What do you need to know about the agents?
↪");

            if (q2.contains("Breach"))
            {
                printAgentDetails(Breach);
            }
            else if (q2.contains("Brimstone"))
            {
                printAgentDetails(Brimstone);
            }
            else if (q2.contains("Cypher"))
            {
                printAgentDetails(Cypher);
            }
            else if (q2.contains("Omen"))
            {
                printAgentDetails(Omen);
            }
            else if (q2.contains("Killjoy"))
            {
                printAgentDetails(Killjoy);
            }
            else if (q2.contains("Raze"))
```

```
            {
                printAgentDetails(Raze);
            }
            else if (q2.contains("Sova"))
            {
                printAgentDetails(Sova);
            }
            else if (q2.contains("Jett"))
            {
                printAgentDetails(Jett);
            }
            else if (q2.contains("Sage"))
            {
                printAgentDetails(Sage);
            }
            else if (q2.contains("Phoenix"))
            {
                printAgentDetails(Phoenix);
            }
            else if (q2.contains("Reyna"))
            {
                printAgentDetails(Reyna);
            }
            else if (q2.contains("Viper"))
            {
                printAgentDetails(Viper);
            }
            else if (q2.contains("Skye"))
            {
                printAgentDetails(Skye);
            }
            else
            {
                System.out.println("I have not heard of the agent.");
            }
        }
        System.out.println("Goodbye.");
        return;
    }
```

Testing

[7]:
```
gameDetails();
```

2.5.4 createAnswer

What it does This method creates a record of each agent.

Implementation (how it works) Creates a record for agent origin, healing capability and ultimate points.

[10]:
```
public static Agent createAnswer(String given_origin, boolean given_heal, int␣
↪given_ultimate)
   {
        Agent a = new Agent();

        a.origin = given_origin;
        a.heal = given_heal;
        a.ultimate = given_ultimate;

        return a;
   }
```

Testing

[7]:
```
createAnswer();
```

2.5.5 printAgentDetails

What it does This method prints the message regarding the desired agent.

Implementation (how it works) This method takes a String argument which is the message to print. It prints that message and returns. The cursor is left on the next line for anything subsequently printed.

[11]:
```
public static void printAgentDetails(Agent a)
{
    System.out.println("The agent is from " + a.origin + ".");
    if(a.heal)
    {
        System.out.println("The agent can heal themselves/their teammates.");
    }
    else
    {
        System.out.println("The agent cannot heal themselves/their teammates.␣
↪");
    }

    System.out.println("They require " + a.ultimate + " points to use their␣
↪ultimate.");
    System.out.println();

    return;
}
```

Testing

[13]:
```
printAgentDetails(a);
```

2.5.6 Running the program

Run the following call to simulate running the complete program.

[ ]:
```
gameDetails();
System.exit(0);
```

```
How many questions do you have?
1
What do you need to know about the agents?
Reyna
The agent is from Mexico.
The agent can heal themselves/their teammates.
They require 6 points to use their ultimate.

Goodbye.
```

2.6 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[ ]:
```
/* ****************************************
@author Anis Faqiehah binti Zainudin
@version 1
@date 1 Nov 2020

A program that decides an agent can decide or not and how many
points for ultimate.
**************************************** */
import java.util.*;

// Create a new type GameQues to hold details of different GameQues
class Agent
{
    String origin;
    boolean heal;
    int ultimate;
}



class QuesaboutGame2
{
    public static void main(String[] p)
```

```
    {

        gameDetails();
        System.exit(0);
    }

    // Create agents details and print the details stored about them.
    //
    public static void gameDetails()
    {
        Agent Breach = createAnswer("Sweden", false, 7);
        Agent Brimstone = createAnswer("United States", false, 6);
        Agent Cypher = createAnswer("Morocco", false, 7);
        Agent Omen = createAnswer("Unknown", false, 7);
        Agent Killjoy = createAnswer("Germany", false, 7);
        Agent Raze = createAnswer("Brazil", false, 6);
        Agent Sova = createAnswer("Russia", false, 7);
        Agent Jett = createAnswer("South Korea", false, 6);
        Agent Sage = createAnswer("China", true, 7);
        Agent Phoenix = createAnswer("United Kingdom", true, 6);
        Agent Reyna = createAnswer("Mexico", true, 6);
        Agent Viper = createAnswer("United States", false, 7);
        Agent Skye = createAnswer("Australia", true, 6);

        final int n = inputInt("How many questions do you have?");

        for(int i = 1; i<=n; i++)
        {
            String q2 = inputString("What do you need to know about the agents?
↪");

            if (q2.contains("Breach"))
            {
                printAgentDetails(Breach);
            }
            else if (q2.contains("Brimstone"))
            {
                printAgentDetails(Brimstone);
            }
            else if (q2.contains("Cypher"))
            {
                printAgentDetails(Cypher);
            }
            else if (q2.contains("Omen"))
            {
                printAgentDetails(Omen);
            }
```

```
            else if (q2.contains("Killjoy"))
            {
                printAgentDetails(Killjoy);
            }
            else if (q2.contains("Raze"))
            {
                printAgentDetails(Raze);
            }
            else if (q2.contains("Sova"))
            {
                printAgentDetails(Sova);
            }
            else if (q2.contains("Jett"))
            {
                printAgentDetails(Jett);
            }
            else if (q2.contains("Sage"))
            {
                printAgentDetails(Sage);
            }
            else if (q2.contains("Phoenix"))
            {
                printAgentDetails(Phoenix);
            }
            else if (q2.contains("Reyna"))
            {
                printAgentDetails(Reyna);
            }
            else if (q2.contains("Viper"))
            {
                printAgentDetails(Viper);
            }
            else if (q2.contains("Skye"))
            {
                printAgentDetails(Skye);
            }
            else
            {
                System.out.println("I have not heard of the agent."); //unable␣
↪to detect agent
            }
        }
        System.out.println("Goodbye.") // exit program
        return;
    }
```

```
   public static Agent createAnswer(String given_origin, boolean given_heal,␣
↪int given_ultimate) //initialize record
   {
       Agent a = new Agent();

       a.origin = given_origin;
       a.heal = given_heal;
       a.ultimate = given_ultimate;

       return a;
   }

   // Print all the known details about an agent given an agent record.
   //
   public static void printAgentDetails(Agent a)
   {
       System.out.println("The agent is from " + a.origin + ".");
       if(a.heal) //using if statement to decide an agent can heal or not
       {
           System.out.println("The agent can heal themselves/their teammates.
↪");
       }
       else
       {
           System.out.println("The agent cannot heal themselves/their␣
↪teammates. ");
       }

       System.out.println("They require " + a.ultimate + " points to use their␣
↪ultimate.");
       System.out.println();

       return;
   }


   public static String inputString (String message)
   {
      String answer;
      Scanner scanner = new Scanner(System.in);

      System.out.println(message);
      answer = scanner.nextLine();

      return answer;
   } // END inputString
```

```
    public static int inputInt (String message)
    {
        String answer = inputString(message);

        return Integer.parseInt(answer);
    } // END inputInt

}
```

END OF LITERATE DOCUMENT

MiniProjectLevel4

December 8, 2020

1 Mini Project(Help bot)

2 Level 4

2.1 Anis Faqiehah binti Zainudin

2.2 1 November 2020

2.3 Version 1

2.4 Summary of the Question

Includes Arrays

Defines and uses accessor methods (all defined in the main class) to access records.

Defines and uses methods including at least one that is passed and uses array arguments

Methods individually commented about what they do and all clearly indented.

2.5 The literate program development

[1]:
```
class ProTeam
{
    String proplayers;
}
```

2.5.1 InputString

What it does This method takes a String question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) It sets up a local scanner to the method, linked to the keyboard. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole String up to ENTER as the response). Whatever typed is stored in a String, and returned as the result of the method.

[17]:
```
public static String input(String message)
{
       String answer;
       Scanner scanner = new Scanner(System.in);
```

```
        System.out.println(message);
        answer = scanner.nextLine();

        return answer;
} // END inputString
```

Testing

[18]:
```
String square = input("What size square of numbers do you want?");
```

```
What size square of numbers do you want?
2
```

2.5.2 InputInt

What it does This method takes an integer question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) The first variable creates a scanner to allow keyboard input. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole integer up to ENTER as the response). Whatever typed is stored in a integer, and returned as the result of the method.

[4]:
```
// input an integer
//
public static int inputInt (String message)
{
        String answer = inputString(message);

        return Integer.parseInt(answer);
}
```

Testing

[5]:
```
int number = inputInt("How old are you?");
```

```
How old are you?
20
```

2.5.3 setPlayer getPlayer

What it does This method creates a record of each player. A get and set method.

Implementation (how it works) Using getter and setter method embedded in the record. The setPlayer stores the value. while getPlayer prints the output in a method that is used later on.

[7]:
```
public static ProTeam setPlayer(ProTeam prt, String ply)
{
```

```
    prt.proplayers = ply;

    return prt;
}

public static String getPlayer(ProTeam prt)
{
    return prt.proplayers;
}
```

Testing

[ ]:
```
String p = setPlayer(p);
String playerteam = getPlayer(p);
```

2.5.4 printStatement

What it does This method creates a record of each pro player team.

Implementation (how it works) Using an if statement and getPlayer method, each team will represent a player.

[20]:
```
public static void printStatement(ProTeam p)
    {
        String playerteam = getPlayer(p);

        if(playerteam.contains("G2 Esports"))
        {
            System.out.println("That team is known from  their player,davidp␣
↪and won 7 Valorant tournaments in September 2020.");
        }
        else if(playerteam.contains("Sentinels"))
        {
            System.out.println("That team is known by their player, Sinatraa␣
↪who was an MVP in one of the tournaments.");
        }
        else if(playerteam.contains("TSM"))
        {
            System.out.println("This team is known by their star player,␣
↪Wardell who is bad at playing rifles but best in handling snipers.");
        }
        else if(playerteam.contains("Cloud9"))
        {
            System.out.println("This team is known by their star player, TenZ␣
↪who is placed first in North America region Valorant.");
        }
        else
```

```
        {
            System.out.println("I don't know about that team.");
        }

    }
```

Testing

[ ]:
```
printStatement(p);
```

2.5.5 createMapArray

What it does This method create array of maps.

Implementation (how it works) This method initialises a string that contains maps and fill them in with the information.

[12]:
```
public static String[] createmapArray()
{
    String [] names_of_maps = {"Ascent", "Bind", "Haven", "Icebox", "Split"};

    return names_of_maps;
}
```

Testing

[ ]:
```
createmapArray();
```

2.5.6 sitesNumber

What it does This method prints the message regarding the number of sites contained in a map.

Implementation (how it works) Using the array names_of_mapes, for certain maps going to have 2 sites and for one map has 3 sites. These method is embedded inside an if statement.

[14]:
```
public static void sitesNumber(String list, String [] names_of_maps)
{
    System.out.println("There are:" + ""); //prints out all the details 0 the␣
↪sites contained in a map

    if (list.contains(names_of_maps[0]) | list.contains(names_of_maps[1]) |␣
↪list.contains(names_of_maps[3]) | list.contains(names_of_maps[4]))
  {
      System.out.println("2 sites, A and B.");
  }
  else if (list.contains(names_of_maps[2]))
  {
    System.out.println("3 sites, A, B and C.");
```

```
    }
    else
    {
        return;
    }
    return;
}
```

Testing

[ ]:
```
sitesNumber(question, sites);
```

2.5.7 createPlayerQues

What it does This method creates a question that stores question about pro team using the accessor method setPlayer.

Implementation (how it works) This method takes a String argument which is the message to print. It prints that message and returns. The cursor is left on the next line for anything subsequently printed.

[15]:
```
public static ProTeam createPlayerQues ()
{
    ProTeam p = new ProTeam();
    String plyques = input("\nWhich pro team do you want to know about?");
    p = setPlayer(p, plyques);     // USE THIS WHEN PRINTING OVERALL STATEMENT
    return p;
}
```

Testing

[ ]:
```
createPlayerQues();
```

2.5.8 Running the program

Run the following call to simulate running the complete program.

[21]:
```
String [] sites = createmapArray();
String question = input("Which map do you need to know how many sites are there?
 ↪");
sitesNumber(question, sites);

ProTeam p = new ProTeam();
p = createPlayerQues();
printStatement(p);
System.out.println("\nUh oh, I'm dying. Gotta recharge before I answer more␣
 ↪questions. See ya!");
```

```
Which map do you need to know how many sites are there?
Haven
There are:
3 sites, A, B and C.

Which pro team do you want to know about?
TSM
This team is known by their star player, Wardell who is bad at playing rifles
but best in handling snipers.

Uh oh, I'm dying. Gotta recharge before I answer more questions. See ya!
```

2.6 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[ ]:
```java
/* ****************************************
@author Anis Faqiehah binti Zainudin
@version 1
@date 1 Nov 2020

A program that asks about the details of the sites and contains the
details of pro players.
**************************************** */

import java.util.*;

class mini
{
        public static void main(String[] param)
        {
    String [] sites = createmapArray();
    String question = input("Which map do you need to know how many sites are␣
 ↪there?"); // ask about map
    sitesNumber(question, sites);

    ProTeam p = new ProTeam(); //create record
    p = createPlayerQues();
    printStatement(p); // print the answer for pro player
    System.out.println("\nUh oh, I'm dying. Gotta recharge before I answer more␣
 ↪questions. See ya!");
        System.exit(0);
    }


    public static void printStatement(ProTeam p) //print statement for pro␣
 ↪players and the team they're representing
```

```
    {
        String playerteam = getPlayer(p);

        if(playerteam.contains("G2 Esports"))
        {
            System.out.println("That team is known from  their player,davidp␣
↪and won 7 Valorant tournaments in September 2020.");
        }
        else if(playerteam.contains("Sentinels"))
        {
            System.out.println("That team is known by their player, Sinatraa␣
↪who was an MVP in one of the tournaments.");
        }
        else if(playerteam.contains("TSM"))
        {
            System.out.println("This team is known by their star player,␣
↪Wardell who is bad at playing rifles but best in handling snipers.");
        }
        else if(playerteam.contains("Cloud9"))
        {
            System.out.println("This team is known by their star player, TenZ␣
↪who is placed first in North America region Valorant.");
        }
        else
        {
            System.out.println("I don't know about that team.");
        }

    }

    public static String[] createmapArray() //maps in an array
    {
        String [] names_of_maps = {"Ascent", "Bind", "Haven", "Icebox",␣
↪"Split"};

        return names_of_maps;
    }

    public static void sitesNumber(String list, String [] names_of_maps) //
↪number of sites contained in a map
    {
        System.out.println("There are:" + ""); //prints out all the details of␣
↪the sites contained in a map

        if (list.contains(names_of_maps[0]) | list.contains(names_of_maps[1]) |␣
↪list.contains(names_of_maps[3]) | list.contains(names_of_maps[4]))
```

```
        {
            System.out.println("2 sites, A and B.");
        }
        else if (list.contains(names_of_maps[2]))
        {
          System.out.println("3 sites, A, B and C.");
        }
        else
        {
            return;
        }
        return;
    }

    public static ProTeam createPlayerQues () // question about pro team
    {
        ProTeam p = new ProTeam();
        String plyques = input("\nWhich pro team do you want to know about?");
        p = setPlayer(p, plyques);     // USE THIS WHEN PRINTING OVERALL␣
↪STATEMENT
        return p;
    }

    public static ProTeam setPlayer(ProTeam prt, String ply) //setter method
    {
        prt.proplayers = ply;

        return prt;
    }

    public static String getPlayer(ProTeam prt) //getter method
    {
        return prt.proplayers;
    }

         public static String input(String message) //input
    {
        Scanner scanner = new Scanner(System.in);
        String answer;

        System.out.println(message);
        answer = scanner.nextLine();

        return answer;
    }

         public static int inputInt(String message) //integer input
```

```
    {
       return Integer.parseInt(input(message));
    }


}

class ProTeam //class
{
    String proplayers;
}
```

END OF LITERATE DOCUMENT

MiniProjectLevel5

December 8, 2020

1 Mini Project(Help bot)

2 Level 5

2.1 Anis Faqiehah binti Zainudin

2.2 1 Dec 2020

2.3 Version 1

2.4 Summary of the Question

Includes Loops within loops. Includes procedural programming style abstract data type with a clearly commented/specified set of operations. Excellent style over comments, indentation, variable usage etc. Methods individually commented and well indented. Clear structure of multiple methods doing distinct jobs that take arguments and return results

2.5 The literate program development

[1]:
```
class Valorant
{
   String agents; // agents' names
   String title; //agents' titles
}
```

2.5.1 InputString

What it does This method takes a String question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) It sets up a local scanner to the method, linked to the keyboard. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole String up to ENTER as the response). Whatever typed is stored in a String, and returned as the result of the method.

[2]:
```
public static String inputString (String message)
{
       String answer;
       Scanner scanner = new Scanner(System.in);
```

```
        System.out.println(message);
        answer = scanner.nextLine();

        return answer;
} // END inputString
```

Testing

[3]:
```
String square = inputString("What size square of numbers do you want?");
```

```
What size square of numbers do you want?
2
```

2.5.2 InputInt

What it does This method takes an integer question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) The first variable creates a scanner to allow keyboard input. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole integer up to ENTER as the response). Whatever typed is stored in a integer, and returned as the result of the method.

[4]:
```
// input an integer
//
public static int inputInt (String message)
{
        String answer = inputString(message);

        return Integer.parseInt(answer);
}
```

Testing

[5]:
```
int number = inputInt("How old are you?");
```

```
How old are you?
20
```

2.5.3 ADT

Unraveling each method

- createAgent This method creates a record of each Agent.
- setAgent A setter method of agent name.
- setTitle A setter method for agent title.
- getAgent A getter method of agent name.

- getTitle A getter method for agent title.

Implementation (how it works) Using setter and getter methods.

[6]:
```
/*  ********************************************************************

    Define an abstract data type of Valorant with operations:
      - create agent and print their title
      - setter and getter methods for agent and title

*/
    public static Valorant createAgent(String agent_name, String agent_title)
    {
        Valorant v = new Valorant();
        setAgent(v, agent_name);
        setTitle(v, agent_title);
        return v;
    }

    public static Valorant setAgent(Valorant v, String ag)
    {
        v.agents = ag;

        return v;
    }

    public static Valorant setTitle(Valorant v, String title1)
    {
        v.title = title1;

        return v;
    }

    public static String getAgent (Valorant v)
    {
     return v.agents;
    }

   public static String getTitle (Valorant v)
    {
     return v.title;
    }
```

Testing

[7]:
```
Valorant v = new Valorant();
setAgent(v, agent_name);
setTitle(v, agent_title);
```

```
String given_title = getTitle(v);
String given_name = getAgent(v);
```

2.5.4 print

What it does It prints the agent and its title.

Implementation (how it works) Using the getter method, initialise it against a given title and given name string and print out a statement.

[8]:
```
public static void print(Valorant v)
{
    String given_title = getTitle(v);
    String given_name = getAgent(v);
    System.out.println("Agent " + given_name + " has the title " + given_title␣
 ↪+ ".");
}
```

Testing

[ ]:
```
print(Breach);
```

2.5.5 AgentQues

What it does This method creates a record of each agent and their titles.

Implementation (how it works) This method uses a nested loop that in a for loop also with a while loop.

[9]:
```
public static void AgentQues()
    {
        Valorant Breach = createAgent("Breach", "Initiator");     //lists of␣
 ↪agents and their titles
        Valorant Brimstone = createAgent("Brimstone", "Controller");
        Valorant Cypher = createAgent("Cypher", "Sentinel");
        Valorant Omen = createAgent("Omen", "Controller");
        Valorant Killjoy = createAgent("Killjoy", "Sentinel");
        Valorant Raze = createAgent("Raze", "Duelist");
        Valorant Sova = createAgent("Sova", "Initiator");
        Valorant Jett = createAgent("Jett", "Duelist");
        Valorant Sage = createAgent("Sage", "Sentinel");
        Valorant Phoenix = createAgent("Phoenix", "Duelist");
        Valorant Reyna = createAgent("Reyna", "Duelist");
        Valorant Viper = createAgent("Viper", "Controller");
        Valorant Skye = createAgent("Skye", "Initiator");
```

```
        final int n = inputInt("How many questions do you have?");

        for(int i = 1; i<=n; i++)
        {
            String q2 = inputString("Which agent do you want to know about?"); /
↪/ using if statement

            if (q2.contains("Breach"))
            {
                print(Breach);
            }
            else if (q2.contains("Brimstone"))
            {
                print(Brimstone);
            }
            else if (q2.contains("Cypher"))
            {
                print(Cypher);
            }
            else if (q2.contains("Omen"))
            {
                print(Omen);
            }
            else if (q2.contains("Killjoy"))
            {
                print(Killjoy);
            }
            else if (q2.contains("Raze"))
            {
                print(Raze);
            }
            else if (q2.contains("Sova"))
            {
                print(Sova);
            }
            else if (q2.contains("Jett"))
            {
                print(Jett);
            }
            else if (q2.contains("Sage"))
            {
                print(Sage);
            }
            else if (q2.contains("Phoenix"))
            {
                print(Phoenix);
            }
```

```
            else if (q2.contains("Reyna"))
            {
                print(Reyna);
            }
            else if (q2.contains("Viper"))
            {
                print(Viper);
            }
            else if (q2.contains("Skye"))
            {
                print(Skye);
            }
            else
            {
                System.out.println("I have not heard of the agent.");
                String ques5 = "y";
                ques5 = inputString("Are you satisfied with the answer given?␣
↪(y/n)");
                while (ques5.equals("n")) // if answer equals to no, rerun the␣
↪program again
                {
                    System.out.println("Maybe try again.");
                    AgentQues();
                    ques5 = inputString("Are you satisfied with the answer␣
↪given? (y/n)");
                }
                System.out.println("Thank you for using my service."); // last␣
↪message
            }
        }
    }
```

2.5.6 Running the program

Run the following call to simulate running the complete program.

[10]:
```
AgentQues();
```

```
How many questions do you have?
1
Which agent do you want to know about?
Reyna
Agent Reyna has the title Duelist.
```

2.6 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[ ]:
```
/*
    Anis Faqiehah binti Zainudin
    DATE 1 Dec 2020
    VERSION 1

  Using a nested loop + ADT for Valorant helpbot.

*/
import java.util.*;

class endofworld
{
    public static void main(String [] param)
    {
        AgentQues();

        System.exit(0);
    }

    public static void AgentQues() // list of agents and their titles
    {
        Valorant Breach = createAgent("Breach", "Initiator");
        Valorant Brimstone = createAgent("Brimstone", "Controller");
        Valorant Cypher = createAgent("Cypher", "Sentinel");
        Valorant Omen = createAgent("Omen", "Controller");
        Valorant Killjoy = createAgent("Killjoy", "Sentinel");
        Valorant Raze = createAgent("Raze", "Duelist");
        Valorant Sova = createAgent("Sova", "Initiator");
        Valorant Jett = createAgent("Jett", "Duelist");
        Valorant Sage = createAgent("Sage", "Sentinel");
        Valorant Phoenix = createAgent("Phoenix", "Duelist");
        Valorant Reyna = createAgent("Reyna", "Duelist");
        Valorant Viper = createAgent("Viper", "Controller");
        Valorant Skye = createAgent("Skye", "Initiator");

        final int n = inputInt("How many questions do you have?");

        for(int i = 1; i<=n; i++) //using a for loop to get exactly what the␣
 ↪user wants
        {
            String q2 = inputString("Which agent do you want to know about?");

            if (q2.contains("Breach"))
            {
                print(Breach);
            }
            else if (q2.contains("Brimstone"))
```

```
            {
                print(Brimstone);
            }
            else if (q2.contains("Cypher"))
            {
                print(Cypher);
            }
            else if (q2.contains("Omen"))
            {
                print(Omen);
            }
            else if (q2.contains("Killjoy"))
            {
                print(Killjoy);
            }
            else if (q2.contains("Raze"))
            {
                print(Raze);
            }
            else if (q2.contains("Sova"))
            {
                print(Sova);
            }
            else if (q2.contains("Jett"))
            {
                print(Jett);
            }
            else if (q2.contains("Sage"))
            {
                print(Sage);
            }
            else if (q2.contains("Phoenix"))
            {
                print(Phoenix);
            }
            else if (q2.contains("Reyna"))
            {
                print(Reyna);
            }
            else if (q2.contains("Viper"))
            {
                print(Viper);
            }
            else if (q2.contains("Skye"))
            {
                print(Skye);
            }
```

```
            else
            {
                System.out.println("I have not heard of the agent.");
                String ques5 = "y";
                ques5 = inputString("Are you satisfied with the answer given?␣
↪(y/n)");
                while (ques5.equals("n")) //using while loop if user not␣
↪satisfied with answer
                {
                    System.out.println("Maybe try again.");
                    AgentQues();
                    ques5 = inputString("Are you satisfied with the answer␣
↪given? (y/n)");
                }
                System.out.println("Thank you for using my service.");
            }
        }
    }



/*  ********************************************************************

    Define an abstract data type of Valorant with operations:
      - create agent and print their title
      - setter and getter methods for agent and title

*/
    public static Valorant createAgent(String agent_name, String agent_title)
    {
        Valorant v = new Valorant();
        setAgent(v, agent_name);
        setTitle(v, agent_title);
        return v;
    }

    public static Valorant setAgent(Valorant v, String ag)
    {
        v.agents = ag;

        return v;
    }

    public static Valorant setTitle(Valorant v, String title1)
    {
        v.title = title1;
```

```
        return v;
    }

    public static String getAgent (Valorant v)
    {
     return v.agents;
    }

    // get title from agent
   public static String getTitle (Valorant v)
    {
     return v.title;
    }

    public static void print(Valorant v) // answer format
    {
        String given_title = getTitle(v);
        String given_name = getAgent(v);
        System.out.println("Agent " + given_name + " has the title " +␣
↪given_title + ".");
    }

    public static String inputString (String message)
    {
       String answer;
       Scanner scanner = new Scanner(System.in);

       System.out.println(message);
       answer = scanner.nextLine();

       return answer;
    } // END inputString

    public static int inputInt (String message)
    {
       String answer = inputString(message);
       return Integer.parseInt(answer);
    }
}

class Valorant
{
   String agents; // agents' names
   String title; //agents' titles
}
```

END OF LITERATE DOCUMENT

MiniProjectLevel6

December 8, 2020

1 Mini Project(Help bot)

2 Level 6

2.1 Anis Faqiehah binti Zainudin

2.2 1 Dec 2020

2.3 Version 1

2.4 Summary of the Question

All the constructs and features above AND Includes a sort algorithm as a separate method. Excellent style over comments, indentation, variable usage etc. Consistent use of well-placed variable names that give a clear indication of their use (perhaps making comments about them redundant). Use of final variables for literal constants. Excellent use of methods throughout Variable declarations are within blocks to reduce scope to a minimum – no use of global variables.

2.5 The literate program development

2.5.1 InputString

What it does This method takes a String question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) It sets up a local scanner to the method, linked to the keyboard. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole String up to ENTER as the response). Whatever typed is stored in a String, and returned as the result of the method.

[1]:
```
public static String inputString (String message)
{
       String answer;
       Scanner scanner = new Scanner(System.in);

       System.out.println(message);
       answer = scanner.nextLine();

       return answer;
} // END inputString
```

Testing

[2]:
```
String square = inputString("What size square of numbers do you want?");
```

```
What size square of numbers do you want?
2
```

2.5.2 sort

What it does This method uses a bubble sort to sort things in ascending manner.

Implementation (how it works) Using for loops and knowing the length of the array, compare the array at first position also the next position and make comparison in alphabetical order.

[4]:
```
public static void sort (String[] array)
{
    boolean sorted=false;
    String temp;
    while (!sorted)
    {
       // array potentially sorted
        sorted = true;

       //traverse array to end switching ill-ordered pairs
       for (int j = 0; j < array.length; j++)
       {
        for (int i = j + 1; i < array.length; i++)
        {
      // comparing adjacent strings
            if (array[i].compareTo(array[j]) < 0)
                {
                temp = array[j];
                array[j] = array[i];
                array[i] = temp;
                }
        }
       }
    }
}
```

Testing

[5]:
```
sort(array);
```

2.5.3 listArray

What it does This method lists the array at random and in order.

Implementation (how it works) Using final int size of game weapons. Printing out the array in random manner firstly. And then using the method sort(array) to get the array sorted alphabetically. Then, printing out the sorted array.

[8]:
```
public static void listArray (String[] array)
{
    final int size = 13;
    System.out.println("List of weapons at random:");
    // print it out
    for (int i=0; i<size; i++)
    {
       System.out.print(array[i]+" \n");
    }
    System.out.println();

    sort(array);

    System.out.println("List of weapons after in order:");
    // print it out again

    for (int i=0; i<size; i++)
    {
       System.out.print(array[i]+" \n");
    }
    System.out.println();
    return;
}
```

Testing

[5]:
```
listarray(array);
```

2.5.4 askQues5

What it does This method asks if the user wants the helpbot to list all the game weapons.

Implementation (how it works) Asks the user for a y/n input and if yes, the helpboy will go through the previous method. If n, then the program will exit.

[6]:
```
public static void askQues5 (String[] array1)
{
    String ques5 = inputString("Would you like to use me to list all the␣
 ↪weapons randomly and in order? (y/n)");
    if (ques5.equals("y"))
    {
        listArray(array1);
    }
    else if (ques5.contains("n"))
```

```
    {
        System.out.println("OK. Bye Bye!");
    }
    else
    {
        return;
    }
    return;
}
```

Testing

[5]:
```
askQues5();
```

2.5.5 Running the program

Run the following call to simulate running the complete program.

[9]:
```
String array[] = {"Classic","Shorty","Frenzy","Ghost","Sheriff",
                  "Stinger", "Specter", "Bucky", "Judge", "Bulldog",
                  "Guardian", "Phantom", "Vandal", "Marshal", "Operator",␣
 ↪"Ares", "Odin"};
askQues5(array);
```

```
Would you like to use me to list all the weapons randomly and in order? (y/n)
y
List of weapons at random:
Classic
Shorty
Frenzy
Ghost
Sheriff
Stinger
Specter
Bucky
Judge
Bulldog
Guardian
Phantom
Vandal

List of weapons after in order:
Ares
Bucky
Bulldog
Classic
Frenzy
Ghost
```

```
Guardian
Judge
Marshal
Odin
Operator
Phantom
Sheriff
```

2.6 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[ ]:
```
/* ****************************************
@author Anis Faqiehah binti Zainudin
@version 1
@date 1 Dec 2020

A program that uses bubblesort to sort weapon names
in Valorant alphabetically.
**************************************** */
//Bubble sort program
import java.util.*;

class mini6
{
    public static void main (String param[])
    {
        // declare array and initialise with random elements
        String array[] = {"Classic","Shorty","Frenzy","Ghost","Sheriff",
                          "Stinger", "Specter", "Bucky", "Judge", "Bulldog",
                          "Guardian", "Phantom", "Vandal", "Marshal",␣
 ↪"Operator", "Ares", "Odin"};
        askQues5(array);

    }

    public static void sort (String[] array)
    {
        boolean sorted=false;
        String temp;
        while (!sorted)
        {
               // array potentially sorted
            sorted = true;

               //traverse array to end switching ill-ordered pairs
```

```
          for (int j = 0; j < array.length; j++)
          {
           for (int i = j + 1; i < array.length; i++)
           {
         // comparing adjacent strings
               if (array[i].compareTo(array[j]) < 0)
                   {
                   temp = array[j];
                   array[j] = array[i];
                   array[i] = temp;
                   }
           }
          }
       }

   public static void listArray (String[] array) //listing the game weapons in order
   {
       final int size = 13;
       System.out.println("List of weapons at random:");
       // print it out
       for (int i=0; i<size; i++)
       {
          System.out.print(array[i]+" \n");
       }
       System.out.println();

       sort(array);

       System.out.println("List of weapons after in order:");
       // print it out again

       for (int i=0; i<size; i++)
       {
          System.out.print(array[i]+" \n");
       }
       System.out.println();
           return;
   }

   public static void askQues5 (String[] array1) //asks the user a y/n question
   {
       String ques5 = inputString("Would you like to use me to list all the weapons randomly and in order? (y/n)");
       if (ques5.equals("y"))
       {
           listArray(array1);
```

```
        }
        else if (ques5.contains("n"))
        {
            System.out.println("OK. Bye Bye!");
        }
        else
        {
            return;
        }
            return;
    }

    public static String inputString (String message)
    {
       String answer;
       Scanner scanner = new Scanner(System.in);

       System.out.println(message);
       answer = scanner.nextLine();

       return answer;
    } // END inputString
}
```

END OF LITERATE DOCUMENT

MiniProjectLevel7

December 8, 2020

1 Mini Project - Help bot

2 Level 7

2.1 Anis Faqiehah binti Zainudin

2.2 8 December 2020

2.3 Version 1

2.4 Summary of the Question

Write a program that uses both file input and file output.

2.5 The literate program development

2.5.1 input

What it does This method takes a String question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) It sets up a local scanner to the method, linked to the keyboard. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole String up to ENTER as the response). Whatever typed is stored in a String, and returned as the result of the method.

[1]:
```
public static String input(String message)
    {
       String answer;
       Scanner scanner = new Scanner(System.in);

       System.out.println(message);
       answer = scanner.nextLine();

       return answer;
    }
```

Testing

[6]:
```
String filename = input("Name the file you want to store the array of weapons␣
↪in VALORANT.");
```

```
Name the file you want to store the array of weapons in VALORANT.
anis.txt
```

2.5.2 writeFile

What it does This method uses print writer and file writer to write in the file.

Implementation (how it works) Using printwriter directing it to filewriter and to the filename that the user wants. Create an array that we did in Mini Project level 6. Insert the array one by one in the file.

[2]:
```
public static void writeFile() throws IOException
{
    String filename = input("Name the file you want to store the array of␣
 ↪weapons in VALORANT.");
    PrintWriter outputStream = new PrintWriter(new FileWriter(filename));

    // Create an array with some sample names to store
    String [] names = {"Classic","Shorty","Frenzy","Ghost","Sheriff",␣
 ↪"Stinger", "Specter",
                                "Bucky", "Judge", "Bulldog", "Guardian",␣
 ↪"Phantom", "Vandal", "Marshal",
                                "Operator", "Ares", "Odin"};

    // Store the names from the array in the file, one name per line
    for (int i = 0; i < names.length; i++)
    {
            outputStream.println(names[i]);
    }

    outputStream.close();
}
```

```
Name the file you want to store the array of weapons in VALORANT.
val.txt
```

Testing

[7]:
```
writeFile();
```

```
Name the file you want to store the array of weapons in VALORANT.
weps.txt
```

2.5.3 InputInt

What it does This method takes an integer question to be printed and then waits for the user to type a response to that question, returning the response.

Implementation (how it works) The first variable creates a scanner to allow keyboard input. This makes available the method scanner.nextLine(). The message is printed and it then allows the user to type the response using nextLine (so takes a whole integer up to ENTER as the response). Whatever typed is stored in a integer, and returned as the result of the method.

[4]:
```
public static int inputInt (String message) // Make scanner available
{
    Scanner scanner = new Scanner(System.in);
    String text_input;
    int numberInput;

    System.out.println(message);
    text_input = scanner.nextLine();
    numberInput = Integer.parseInt(text_input); // Change string to integer

    return numberInput; // return variable
}
```

Testing

[8]:
```
int number = inputInt("How many sheeps are there in your house?");
```

```
How many sheeps are there in your house?
0
```

2.5.4 AccessFile

What it does This method asks the user which file does the user want to access and how many items the user wants to list from the file.

Implementation (how it works) The scanner is made available to ask the first question and second question. BufferedReader is made available to read filename2 and read one the items that is asked by the user using for loop.

[10]:
```
public static void AccessFile() throws IOException
{
    Scanner scanner = new Scanner(System.in);
    System.out.println("What is the name of the file you want to access?");
    String filename2 = scanner.nextLine();
    int n = inputInt("How many items you want me to list in the file?(max.␣
 ↪17)");

    BufferedReader inputStream = new BufferedReader (new FileReader(filename2));
    for(int i = 1; i <=n; i++)
    {
    String line = inputStream.readLine();
    System.out.println(line);
    }
```

```
    inputStream.close();
}
```

Testing

[11]:
```
AccessFile();
```

```
What is the name of the file you want to access?
weps.txt
How many items you want me to list in the file?(max. 17)
5
Classic
Shorty
Frenzy
Ghost
Sheriff
```

2.5.5 Running the program

Run the following call to simulate running the complete program.

[12]:
```
writeFile();
AccessFile();
```

```
Name the file you want to store the array of weapons in VALORANT.
test.txt
What is the name of the file you want to access?
test.txt
How many items you want me to list in the file?(max. 17)
2
Classic
Shorty
```

2.6 The complete program

This version will only compile here. To run it copy it into a file called initials.java on your local computer and compile and run it there.

[ ]:
```
/* ****************************************
@author Anis Faqiehah binti Zainudin
@version 1
@date 8 December 2020

A program that uses file input and file output
to access and read file.
**************************************** */
import java.util.*;
import java.io.File;
```

```java
import java.io.IOException;

public class mini7 {
  public static void main(String[] args)
  {
    writeFile();
    AccessFile();
    System.exit(0);
  }

    public static void writeFile() throws IOException // method to create file␣
↪that stores array
    {
        String filename = input("Name the file you want to store the array of␣
↪weapons in VALORANT.");
        PrintWriter outputStream = new PrintWriter(new FileWriter(filename));

        // Create an array with some sample names to store
        String [] names = {"Classic","Shorty","Frenzy","Ghost","Sheriff",␣
↪"Stinger", "Specter",
                                    "Bucky", "Judge", "Bulldog", "Guardian",␣
↪"Phantom", "Vandal", "Marshal",
                                    "Operator", "Ares", "Odin"};

        // Store the names from the array in the file, one name per line
        for (int i = 0; i < names.length; i++)
        {
                outputStream.println(names[i]);
        }

        outputStream.close();
    }

    public static void AccessFile() throws IOException // method to access and␣
↪read file
    {
        Scanner scanner = new Scanner(System.in); // make scanner available
        System.out.println("What is the name of the file you want to access?");
        String filename2 = scanner.nextLine();
        int n = inputInt("How many items you want me to list in the file?(max.␣
↪17)");

        BufferedReader inputStream = new BufferedReader (new␣
↪FileReader(filename2)); // use bufferedreader
        for(int i = 1; i <=n; i++)
        {
```

```
        String line = inputStream.readLine(); // read lines that has been asked␣
↪for
        System.out.println(line);
        }

        inputStream.close();
    }

    public static String input(String message) // input method that returns␣
↪string
    {
        String answer;
        Scanner scanner = new Scanner(System.in);

        System.out.println(message);
        answer = scanner.nextLine();

        return answer;
    }

    public static int inputInt (String message) // Make scanner available
    {
        Scanner scanner = new Scanner(System.in);
        String text_input;
        int numberInput;

        System.out.println(message);
        text_input = scanner.nextLine();
        numberInput = Integer.parseInt(text_input); // Change string to integer

        return numberInput; // return variable
    }
  }
}
```

END OF LITERATE DOCUMENT