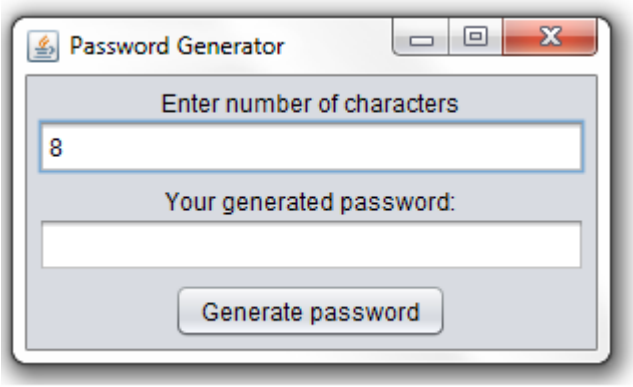
# Section 1(3x8=24 M)

1. Write password generator in GUI. The program shall ask user the length of the password and generate a password of length 8 with random characters.

   Hints:

   - You can use Math.random() to generate a random number in double in the range of [0.0, 1.0) and to select a character randomly from array of alphabets and symbols.




2. Write a Program using Threads for the following case study:
   Library management: Write a simple program showing a typical operation such as issue and return book for Java Books in library via multiple sections (E2-206, E2-207, E2-208). Each section runs via separate thread.
   For issuing a Java book first request the no of copies required, then if sufficient no of copies are available then display the name of the section taking the book, and no of copies issued and available. If sufficient copies are not available, issue the available copies and also show how many copies could not be issued.
   For returning the books, return the same no copies that were issued. After returning display the name of the section returned the book and the available no of copies. Each section will run via separate thread. Set the name of the threads as E2-206, E2-207, E2-208.

   Test the following case: Initially available 100 copies.

```
> E2-206: Requires = 35, Issued = 35, Available = 55  //Sleeps for 2
secs
> E2-206: Requires = 45, Issued = 45, Available = 10  //Sleeps for 2
secs
> E2-206: Requires = 30, Issued = 10, Could not issue=20, Available = 0
//Sleeps 5 secs
> E2-206: Returned = 5, Available = 5
>E2-207: Returned = 10, Available = 15
```

```
>E2-206: Requires = 20, Issued = 15, Available = 0 //Wake up and request
```
Use sleep() method of thread class to simulate the output.

3. Write a program to take input set of pairs {author name, book name} from command line and store all the strings in an array. Perform the following operations on all the strings:
   a. Count no of book names contains the word “Java”
   b. Print all the books by a given author
   c. Sort the books by book name.

```
>java Books “Head First Java,Kathie ” “Java 2,Herbert” “C++,
Herbert” “Thinking in Java,Bruce”
>3 books contains the word java
>Enter author name: Herbert
>Java 2,Herbert
>C++, Herbert
>Book in sorted order:
>C++, Herbert
>Head First Java,Kathie
>Java 2,Herbert
>Thinking in Java,Bruce
```

4. Write a class Cake with one instance variable flavor two constructors, default and parametrized. The default constructor should call the parametrized constructor and pass the default values of flavor “Vanilla”. Add methods bake(), frost() to bake/frost for given time period and toString() method to print info of Cake. Maximum baking time should not exceed 60 minutes. Create a sub class BirthdayCake with extra parameter age with default value 1. Add two constructors, default and parametrized and two methods putCandlesOnCake() and toString().
   Create person class with a prepare() method. The prepare() method should call all the methods based on whether the person is preparing Normal Cake or BirthdayCake.

   Write a Test class and call the prepare() method of Person class to get the following output:

```
>Preparing Cake
>It’s a normal cake with vanilla flavor
>The cake is baking for 20 minutes
>The cake is frosting for 30 minutes.
> Normal cake is prepared
>Preparing BirthdayCake
>It’s a BirthdayCake for 5 year old with Chocolate flavor
> The cake is baking for 30 minutes
>The cake is frosting for 45 minutes.
>Putting 5 candles on the birthday cake
>BirthdayCake is prepared
```

# Section 2 (1x16=16 M)

1. Considered a case study RGUKT Techfest where events could be a Technical and a Cultural. Technical events could be either a Working Model or Virtual reality Gaming. Cultural events could be either a Dancing or Fashion Frenzy. We are going to create an *event* interface representing events such as Technical and Cultural with two methods name() (name of event) and price() (ticket price for event). Two abstract Classes *Technical* and *Cultural* implement *event* Interface. Concreate classes Smart Village and VirtualReality are subclasses of technical, Concreate classes Dncing and Fashion Frenzy are subclasses of Cultural. All the above classes should be present in pakage events.

   Create Ticket Class with methods getOrdinaryTicket()(should return Ordinary Ticket Price), getPremiumTicket(should return Premium Ticket Price), here Ordinary Ticket is combination of VirtualReality games and Dancing events, Premium Ticket is combination of SmartVillage models and Fashion Frenzy events. Take input from user for type of ticket and print price of the ticket. If user entered ticket type is not avilable throw custom exception with message "that type of ticket is not avilable"