MANUAL TECNICO

Apuestas

Introducción

El desarrollo de este software consta de varios pasos, por lo cual es importante estar siempre en contacto con el cliente, para tener información actualizada y de esta forma llevar el proceso de desarrollo de la mejor manera.

Aquí se describirá el proceso de desarrollo, así de temas de suma importancia para los técnicos que deseen conocer mas acerca del programa, por lo cual se adjuntan una serie de datos que le servirán para su conocimiento.

Objetivo

Otorgar soporte a los técnico o programadores que quieran conocer el proceso de desarrollo de la app.

Datos Técnicos

- Sistema Operativo Windows, Linux
- Minimo 1Gb de ram
- Lenguajes de programación utilizados Java

Análisis de complejidad de cada servicio critico

Ingreso de datos

public void insertBeginning(Gambler bet){

```
        Nodo nodo = new Nodo(bet);
        System.out.println("Nodo creado, Nombre: "+ nodo.bet.getName());

        nodo.next = head;
        head = nodo;
        longitude ++;
    }
```

Complejidad O(1)

Verificación de apuestas

```
public void verifyBets(){
        Gambler[] datos2 = listOfBets.getBets();
```

```
        for(int i=0; i<datos2.length; i++){
            boolean [] repetidos = {false, false, false, false, false, false, false, false,
false, false};
            int cambios =0;
            System.out.println("\n\nverificando a: "+ datos2[i].getName());
            for(int j=0 ; j<datos2[i].getBets().length; j++){
                //System.out.println("revisando el array"+
Arrays.toString(datos2[i].getBets()));
                try {
                    int n = datos2[i].getBets()[j];
                    //System.out.println("verificando el numero del array
"+datos2[i].getBets()[j] );
                    //System.out.println(Arrays.toString(repetidos));
                    String msg = "Error, el numero ya existia";
                    if(n>0 && n<=10){
                        System.out.println("entre aqui, vericando el numero: "+ n);
                        switch (n){
                            case 1: if(!repetidos[0]){repetidos[0] = true;
cambios++;}else{System.out.println(msg);}break;
                            case 2:if(!repetidos[1]){repetidos[1] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 3:if(!repetidos[2]){repetidos[2] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 4:if(!repetidos[3]){repetidos[3] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 5:if(!repetidos[4]){repetidos[4] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 6:if(!repetidos[5]){repetidos[5] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 7:if(!repetidos[6]){repetidos[6] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 8:if(!repetidos[7]){repetidos[7] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 9:if(!repetidos[8]){repetidos[8] =
true;cambios++;}else{System.out.println(msg);}break;
                            case 10:if(!repetidos[9]){repetidos[9] =
true;cambios++;}else{System.out.println(msg);}break;
                        }

                    }else{System.out.println("El numero esta fuera de rango, por lo
tanto no se guardara");}

                } catch (Exception e) {
                    //TODO: handle exception
```

```
                        System.out.println("\n\nerror en apostador: "+
datos2[i].getName()+" con dato en pos: "+ j+"\n\n");
                    }


                }
                if(cambios ==10){
                    //Guardamos la apuesta osea no hacemos nada :D o si?
                    System.out.println("Gambler: "+ datos2[i].getName()+" debe
guardarse\n\n");
                    acepted.insertBeginning(datos2[i]);
                }else{
                    //borramos la apuesta
                    System.out.println("\n\nnombre a eliminar: "+
datos2[i].getName()+"\n\n");
                    //listOfBets.deleteBet(i);
                    rejected.insertBeginning(datos2[i]);
                }

            }

        }
```

Complejidad O(n) || Complejidad O(n2)

```
public void saveArchive(Gambler[] datos) {

        JFileChooser chooser = new JFileChooser();
        chooser.addChoosableFileFilter(new FileNameExtensionFilter("CSV .csv",
"csv"));
        chooser.setApproveButtonText("Guardar");
        chooser.showSaveDialog(null);
        File archive = new File(chooser.getSelectedFile() + ".csv");
        if (!archive.exists()) {
            try {
                archive.createNewFile();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
        try {
            //FileWriter fileWriter=new FileWriter(archive);
            BufferedWriter br = new BufferedWriter(new FileWriter(archive));
            br.write(getDatos(datos));
            br.close();
        } catch (Exception e) {
            //TODO: handle exception
        }


    }
```

Complejidad O(n)

```
public   void readFile() {

        String line = "";
        String splitBy = ",";
        try {
            BufferedReader br = new BufferedReader(new FileReader(path));
            int cont = 0;
            while ((line = br.readLine()) != null)
            //returns a Boolean value
            {
                boolean save = true;
                String[] bet = line.split(splitBy);
                int[] orderHorses = new int[10];
                try {
                    //use comma as separator
                    orderHorses[0] = Integer.parseInt(bet[2].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[1] = Integer.parseInt(bet[3].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[2] = Integer.parseInt(bet[4].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[3] = Integer.parseInt(bet[5].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[4] = Integer.parseInt(bet[6].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[5] = Integer.parseInt(bet[7].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[6] = Integer.parseInt(bet[8].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[7] = Integer.parseInt(bet[9].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[8] = Integer.parseInt(bet[10].replaceAll("<",
"").replaceAll(">", ""));
                    orderHorses[9] = Integer.parseInt(bet[11].replaceAll("<",
"").replaceAll(">", ""));

                    //System.out.println(" Name=" + bet[0].replaceAll("<",
"").replaceAll(">", "")   + ", Amount=" + bet[1] + ", Winner=" + bet[2] + ", second= "
+ bet[3]);
                } catch (Exception e) {
                    //JOptionPane.showMessageDialog(null, "Error, datos con formato
incorrecto ");
                    System.out.println("encontre un error en la linea: "+cont);
                    save = false;
```

```
                }
                if(save) {
                    op.registraDatos(bet[0].replaceAll("<", "").replaceAll(">", ""),
orderHorses, Integer.parseInt(bet[1].replaceAll("<", "").replaceAll(">", "")));

                }
                cont ++;

            }
            JOptionPane.showMessageDialog(null, "Apuestas agregadas
correctamente");
        }
        catch(IOException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Error, verifique el archivo de
entrada");

        }

  }
```

Complejidad O(n)

Cálculo de los resultados al finalizar la carrera

```
public void verifica() {
        Gambler[] datos = acepted.getBets();
        for(int i =0; i<datos.length; i++) {
            int puntos = 0;
            for(int j =0; j<orderH.length ; j++) {
                switch (j+1){
                    case 1:if(orderH[j]==datos[i].getBets()[j]) {
                        puntos = puntos + 10;
                    } break;
                    case 2:if(orderH[j]==datos[i].getBets()[j]) {
                        puntos = puntos + 9;
                    } break;
                    case 3:if(orderH[j]==datos[i].getBets()[j]) {
```

```
                    puntos = puntos + 8;
                } break;

                case 4:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 7;
                } break;
                case 5:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 6;
                } break;
                case 6:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 5;
                } break;
                case 7:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 4;
                } break;
                case 8:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 3;
                } break;
                case 9:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 2;
                } break;
                case 10:if(orderH[j]==datos[i].getBets()[j]) {
                    puntos = puntos + 1;
                } break;
            }


        }

        datos[i].setPoints(puntos);

    }

}
```

Complejidad O(n) || Complejidad O(n2)

Ordenamiento de los resultados

```
public Gambler[] Insercion () {
        Gambler [] datos = acepted.getBets();
        for (int i=1; i < datos.length; i++) {
           int aux = datos[i].getPoints(); //datos[i]
           Gambler aux2 = datos[i];
           int j;
           for (j=i-1; j >=0 && datos[j].getPoints() > aux; j--){
                 //vector[j+1] = vector[j];
                 datos[j+1] = datos[j];
            }
           //vector[j+1] = aux;
           datos[j+1] = aux2;
        }
        return datos;
    }
```

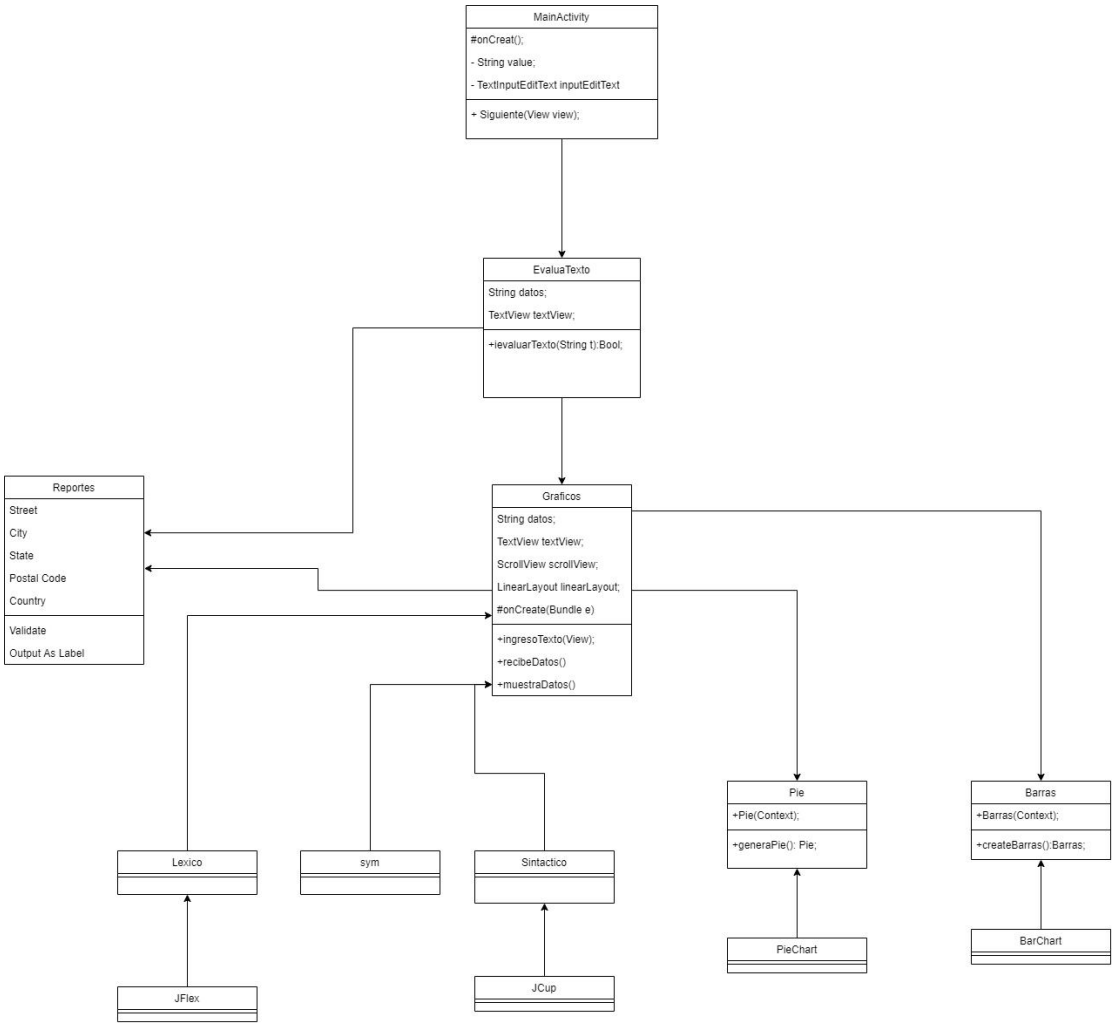
Complejidad O(n2)

Argumentación de los métodos utilizados

Método de inserción

El método de inserción fue utilizado ya que reduce el numero de pasos a realizar, para ordenar un arreglo, a comparación de otros métodos como el caso de burbuja ya que este realiza mas pasos y por lo tanto, tiene un tiempo de respuesta mas lento.

Diagrama de clases



Anexos

Para mas información puede comunicarse al numero de teléfono

+502 xxxxxxxx

O al correo xxxxxx@dominio.com

Sera un gusto atenderle.

Company Inc.