Project 1

<Plus X Puzzle>

CIS-18 86625
Name: Oh, Yoosun
Date: 4/21/2006

Introduction

Title: Plus X Puzzle

This is the number puzzle game.
The problem number is showed on the right upper corner, and the user have to press the number buttons to make the problem number.
If the ‘x2’ or ‘x3’ button is used, the total number will be doubled or tripled.

Ex) If the ‘2’, ‘x2’, ‘1’ and ‘x3’ are selected, the result is ((2*2) + 1) * 3 = 15

When it is impossible to make the showed number, the game is over.
If all number buttons are cleared, the game is solved, and a bonus score is added.

This game can improve children’s calculation ability.
Also even adult can enjoy this game if he try to get higher score by using ‘x2’ and ‘x3’.

Summary

Project size: about 280 lines
The number of variables: about 30
The number of method: 5

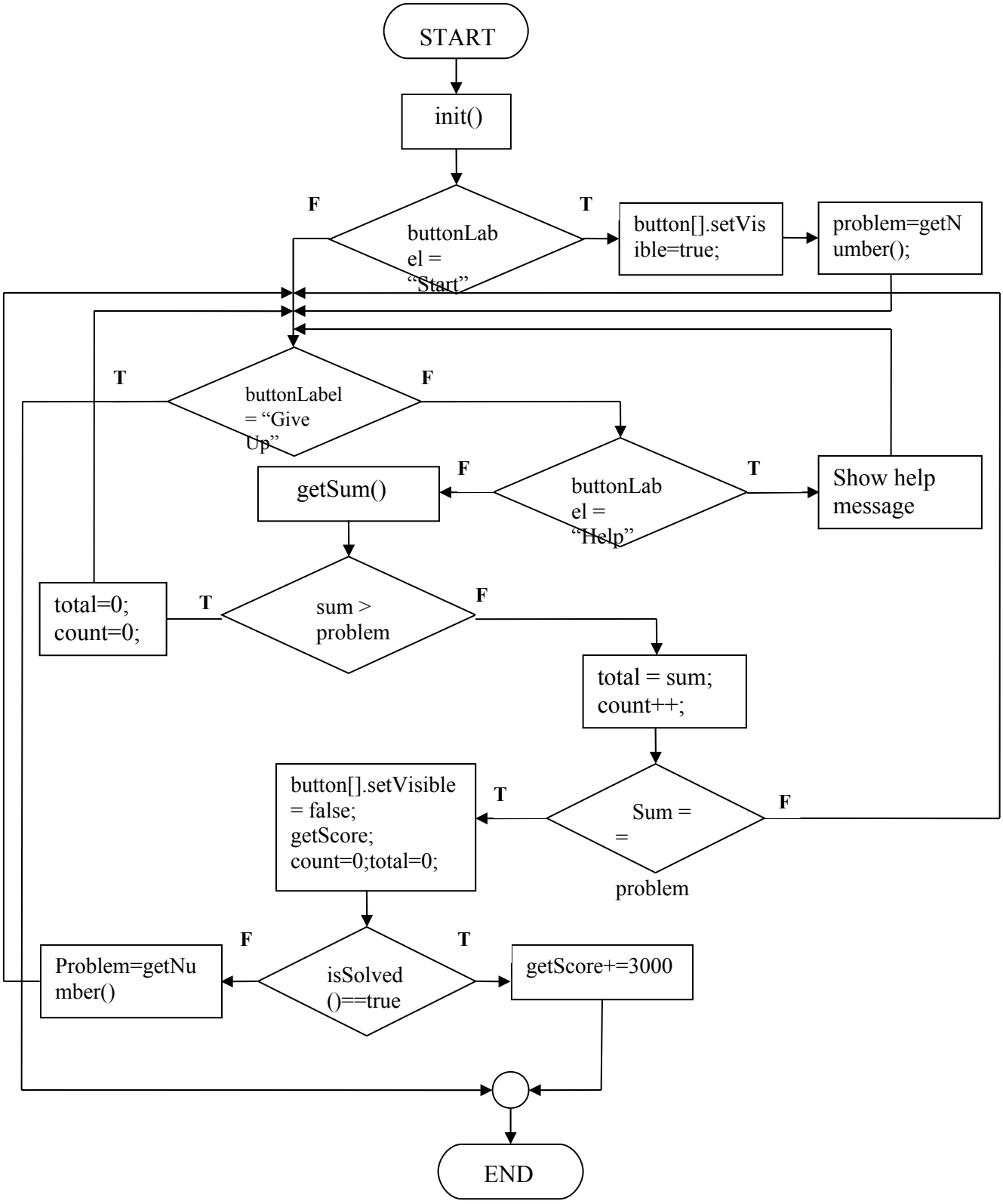
This project includes many concepts that we learned from the chapters in the book
Also, it has many possibilities to be extended for next project.
For example, time thread for extra score point (if it is solved early), adding images etc.

It took almost two weeks because I tried to do many challenging parts that we’ve not learned yet.
I got many troubles and left many of them for next project.
I’m not satisfied with this project, but it was a good experience.
I realized that Java has numerous functions, and it is fun to learn these.
Although this is a very simple game, I think I tried to reflect not only many concepts we have learned but also a few new concepts.
The new concepts that I had to construct from other chapters are array, object oriented programming, layout manager, graphical functions, sound and etc.

Description

The main point that I programmed this project is button handler.
I programmed the action performed as what button is pressed

Flow Chart


START
init()
buttonLabel = "Start"
F
T
button[].setVisible=true;
problem=getNumber();
buttonLabel = "Give Up"
T
F
getSum()
F
buttonLabel = "Help"
T
Show help message
total=0; count=0;
T
sum > problem
F
total = sum; count++;
button[].setVisible = false; getScore; count=0;total=0;
T
Sum = = problem
F
Problem=getNumber()
F
isSolved()==true
T
getScore+=3000
END

Pseudo Code

Initialize

If the start button is pressed
Display puzzle buttons
Get and print a problem number

Else if the Give Up button is pressed
Print the score and finish the game

Else if the Help button is pressed
Show the help message

Else
Call the getSum() function

If sum is greater than problem number
Set count to zero
Set total to zero
Start again
Else
Add one to count
Assign the sum to total

If sum is equal to the problem number
Calculate and print the score
Set count to zero
Set total to zero

If the game is solved
Add 3000 to score
Print the score and finish the game
Else
Start again

Major Variables

Type	Variable Name	Description	Location
Integer	size	The size of puzzle	init()
	MAXNUM	The array size(number[], button[])	init()
	number[]	The array that contains random number 1 to 10 for button label	init(), setButtonColor(int i)
	findNum[]	The array that contains pressed-button number	init(), actionPerformed(ActionEvent e) {when the button[] is pressed}
	problem	The problem number that is got from random function	actionPerformed(ActionEvent e) {when the start button is pressed or the former problem is solved}
	sum	The sum of numbers that is pressed	getSum()
	total	The sum of number after comparing with problem	actionPerformed(ActionEvent e)
	count	The number of pressed-buttons	actionPerformed(ActionEvent e)
	getScore	The score	actionPerformed(ActionEvent e)
JButton	button[]	The JButton array that contains numbered-buttons	init() When the button[] is pressed
	start	The start button	init() When the start button is pressed
	giveUp	The give up button	init() When the giveUp button is pressed
	helpBt	The help button	init() When the help button is pressed
JPanel	buttonPanel	The panel that contains button[]	init()
	startPanel	The panel that contains start and giveUp buttons	init()
JLabel	scoreLabel	The “Score” label	init()
	probLabel	The “Problem” label	init()
JTextField	score	The text field for score	init()
	probNum	The text field for problem	init()
String	help	Help message	init() When the Help button is pressed
	btName	Assign the button label (1 to 10, x2 or x3)	init()
	buttonLabel	Get the pressed-button label	When the button is pressed
Title	titlePanel	The panel that contains title	init(), class Title()
Score	scorePanel	The panel that contains scoreLabel, probLabel, score, and probNum	init(), class Score()
Font	f1, f2	Decide font style and size of scoreLabel, probLabel, score and probNum	init()

Java Constructs

Chapter	New syntax and Keywords	Location
2	public class (class definition)	Public class Project Public class ButtonHandler
	JOptionPane.showInputDialog	sizeInput = JOptionPane.showInputDialog
	JOptionPane.showMessageDialog JOptionPane.INFORMATION_MESSAGE	JOptionPane.showMessageDialog(null, "Your score is " + Integer.toString(getScore), "Score", JOptionPane.INFORMATION_MESSAGE);
	integer.parseInt	Size = Integer.parseInt(sizeInput)
	Import statement Javax.swing package	Iimport javax.swing.*;
	If structure	If (sum == problem) {}
	Equality operators and relational operators (==, !=, >, <, >=, <=)	If (number[i]==10) If (sum > problem)
	Arithmetic operators (+, -, *, /)	size*size Count*30*bonus2*bonus3
	Int primitive type	Int MAXNUM, size…..
	String	String sizeInput, btName, …
	Void keyword	Void init()
3	Extends JApplet	Public class Project extends JApplet
	Appletviewer, HTML tag	
	Init()	Public void init()
	g.drawString	g.srawString("PLUS X PUZZLE", 150, 55);
	Java.awt package	Import java.awt.*;
	Java.awt.event package	Import java.awt.event.*;
4	If/else selection structure	If (buttonLabel == "Start") {} Else {}
	Assignment operator (+=)	getScore += 3000;
	Increment operator (++)	Count++;
	While repetition structure	While (i<MAXNUM) {}
5	For repetition structure	For (int i=0; i<MAXNUM; i++) {}
	JTextArea	helpArea = new JTextArea(10, 10);
	Switch	Switch(number[i]) {}
	Break	Case 1:button[i].setBackground(Color.blue); Break;
	Default	Default: break;
	Case	Case 1: case 2: ….
	Logical operators (true, false)	Button[i].setvisibl(true); probNum.setEditable(false);
6	Container c = getContentPane();	Container c = getContentPane();
	Add	c.add(titlePanel, BorderLayout.NORTH);
	setText	probNum.setText(Integer.toString(problem))

	setEditable	probNum.setEditable(false);
	Return	Return true;
	Math.random()	Number[i] = 1+(int)(Math.random()*10);
	ActionListener	Public class ButtonHandler implements ActionListener
	actionPerformed(ActionEvent e)	Public void actionPerformed(ActionEvent e)

	button = new JButton	Button = new JButton("Start");
		giveUp = new JButton("Give Up");
		helpBt = new JButton("Help");

	Label = new JLabel	scoreLabel = new JLabel("Score"); probLabel = new JLabel("Problem");
	textField = new JTextField	Score = new JTextField(4); probNum = new JTextField(2);
	setLayout	buttonPanel.setLayout(new GridLayout(size,size));
	FlowLayout	Default layout (Title Panel, Score Panel, startPanel)
	addActionListener(this)	Button[i].addActionListener(handler);
	Method	Public int getNumber() Public Boolean isSolved() Public void setButtonColor(int i)
7	Array	findNum = new int[20] number = new int[MAXNUM]
	Font.BOLD, Font.ITALIC, Font.PLAIN	Font f1 = new Font("Arial", Font.BOLD, 22);
	setFont	g.setFont(f);
8	Instance of a class	Title titlePanel = new Title(); Score scorePanel = new Score();
	Integer.toString	btName = Integer.toString(number[i]);
11	drawRoundRect	g.drawRoundRect(10, 250, 110, 110, 20, 20);
	fillRect	g.fillRect(130, 15, 350, 50);
	fillRoundRect	g.fillRoundRect(10, 250, 110, 110, 20, 20);
	clearRect	g.clearRect(135, 20, 350, 50);
	setBackground	Button[i].setBackground(Color.gray);
	Color	buttonPanel.setBackground(Color.white);
12	BorderLayout	c.setLayout(new BorderLayout());
	GridLayout	buttonPanel.setLayout(new GridLayout(size,size));
	Panel = new JPanels()	buttonPanel = new JPanel(); startPanel = new JPanel();
	getActionCommand	buttonLabel = e.getActionCommand();
	getSource	If (e.getSource() == button[i])
	setVisible	Button[i].setVisible(false);
	setEnabled	Start.setEnabled(false);
13	Dimension	Public Dimension getPreferredSize()
	getPreferredSize	titlePanel.setSize(getPreferredSize());

	paintComponent	Public void paintComponent(Graphics g)
	Super.paintComponent(g)	Super.paintComponent(g);
16	Play getCodeBase()	Play(getCodeBase(), “audio/return.au”);

Reference

1. textbook
2. API / Languages (Java 2 Platform API Specification)
3. jdk1.2.1 – demo files

Program

```
// Plus X Puzzle

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Project extends JApplet
{
   private int MAXNUM, size;
   private JButton button[], start, giveUp, helpBt;
   private JPanel buttonPanel, startPanel;
   private JLabel scoreLabel, probLabel;
   private JTextField score, probNum;
   private JTextArea helpArea;
   private String sizeInput, btName, buttonLabel, help;
   private int number[], findNum[], problem, sum,
           total=0, count=0, getScore=0, bonus2=1, bonus3=1;
   private Title titlePanel;
   private Score scorePanel;
   private Font f1, f2, f3;

   public void init()
   {
      // get the size of puzzle
      do {
         sizeInput = JOptionPane.showInputDialog("Please enter the size (2-10)");
         size = Integer.parseInt(sizeInput);
         MAXNUM = size*size;
      } while (size>10 || size<2);
```

```
// the panel that contains puzzle title
titlePanel = new Title();
titlePanel.setBackground(Color.pink);
titlePanel.setSize(getPreferredSize());

// the panel that contains number buttons
buttonPanel = new JPanel();
buttonPanel.setLayout(new GridLayout(size, size));
buttonPanel.setBackground(Color.darkGray);

// the score panel that contains problem and score
scorePanel = new Score();
scorePanel.setSize(getPreferredSize());

// the panel that contains a start, giveUp and help button
startPanel = new JPanel();

// number buttons
findNum = new int[20];
number = new int[MAXNUM];
ButtonHandler handler = new ButtonHandler();
button = new JButton[MAXNUM];

// start button
start = new JButton("Start");
start.addActionListener(handler);

// giveUp button
giveUp = new JButton("Give Up");
giveUp.addActionListener(handler);
giveUp.setEnabled(false);

// text area that contains a help message
helpArea = new JTextArea(10, 10);
helpBt = new JButton("Help");
helpBt.addActionListener(handler);
help = "See the problem number\n   showed on the right upper corner,\n" +
"   and press the number buttons \n   to make the problem number.\n" +
"If the 'x2' or 'x3' button is used, \n" +
"   the total number will be doubled or tripled.\n" +
"When it is impossible to make \n   the showed number, the game is over.";
helpArea.setText(help);
f3 = new Font("Aria", Font.BOLD, 15);
helpArea.setFont(f3);

// score label and text field
```

```
f1 = new Font("Arial", Font.BOLD, 22);
scoreLabel = new JLabel("Score");
scoreLabel.setFont(f1);
score = new JTextField(4);
score.setEditable(false);
score.setFont(f1);

// problem label and text field
f2 = new Font("Arial", Font.BOLD, 50);
probLabel = new JLabel("Problem");
probLabel.setFont(f1);
probNum = new JTextField(2);
probNum.setEditable(false);
probNum.setFont(f2);

// assign the button lables
for (int i=0; i<MAXNUM; i++) {
   number[i] = 1 + (int)(Math.random()*10);
   if (number[i] == 10)
     btName = "x2";
   else if (number[i] == 9)
     btName = "x3";
   else
     btName = Integer.toString(number[i]);

   button[i] = new JButton(btName);
   setButtonColor(i);
   button[i].setVisible(false);
   buttonPanel.add(button[i]);
   button[i].addActionListener(handler);
}

// add objects to panels
scorePanel.add(probLabel);
scorePanel.add(probNum);
scorePanel.add(scoreLabel);
scorePanel.add(score);
startPanel.add(start);
startPanel.add(giveUp);
startPanel.add(helpBt);

Container c = getContentPane();
c.setLayout(new BorderLayout());

// set panels
c.add(titlePanel, BorderLayout.NORTH);
```

```
    c.add(buttonPanel, BorderLayout.CENTER);
    c.add(scorePanel, BorderLayout.EAST);
    c.add(startPanel, BorderLayout.SOUTH);
}

public class ButtonHandler implements ActionListener
{
    public void actionPerformed(ActionEvent e)
    {
        buttonLabel = e.getActionCommand();

        // when a start button is pressed
        if (buttonLabel == "Start") {
            for (int i=0; i<MAXNUM; i++)
              button[i].setVisible(true);
            problem = getNumber();
            probNum.setText(Integer.toString(problem));
            start.setEnabled(false);
            giveUp.setEnabled(true);
        }

        // when a giveUp button is pressed
        else if (buttonLabel == "Give Up") {
            for (int i=0; i<MAXNUM; i++) {
                if (button[i].isVisible() == true){
                  button[i].setEnabled(false);
                  button[i].setBackground(Color.gray);
                }
            }
            total=0; count=0;
            probNum.setText("");
            score.setText("");
            giveUp.setEnabled(false);
            JOptionPane.showMessageDialog
                (null, "Your score is " + Integer.toString(getScore));
        }

        // when a help button is pressed
        else if (buttonLabel == "Help") {
             JOptionPane.showMessageDialog(null, helpArea, "Help",
               JOptionPane.INFORMATION_MESSAGE);
        }

        // when a number button is pressed
        else {
            getSum();    // get the sum
```

```
// when the sum is greater than the problem number
if (sum > problem) {
  play(getCodeBase(), "audio/exceed.au");
  JOptionPane.showMessageDialog(null, "You exceeded the number!");
  for (int i=0; i<count; i++) {
    button[findNum[i]].setEnabled(true);
    setButtonColor(findNum[i]);
  }
  count = 0;
  total = 0;
}
// when the sum is less than or equal to the problem number
else {
  for (int i=0; i<MAXNUM; i++)
    if (e.getSource() == button[i])
      findNum[count] = i;
  play(getCodeBase(), "audio/ding.au");

  button[findNum[count]].setEnabled(false);
  button[findNum[count]].setBackground(Color.gray);
  count++;
  total = sum;

  // when it is solved
  if (sum == problem) {
    play(getCodeBase(), "audio/return.au");

    for (int i=0; i<count; i++)
      button[findNum[i]].setVisible(false);

    getScore += count * 30 * bonus2 * bonus3;
    count=0; total = 0;
    bonus2=1; bonus3=1;
    if (isSolved() == true) {
      getScore += 3000;
      play(getCodeBase(), "audio/spacemusic.au");
      JOptionPane.showMessageDialog(null,
       "Congratulations!!!\nYour score is " + Integer.toString(getScore));
    }
    else {
      problem = getNumber();
      probNum.setText(Integer.toString(problem));
      score.setText(Integer.toString(getScore));
    }
  }
```

```
            }
        }
    }
}

// get the random number for a problem number
public int getNumber()
{
    return (1+(int)(Math.random()*2)) * (1+(int)(Math.random()*10));
}

// get the sum
public void getSum()
{
    if (buttonLabel == "x2") {
        sum = total*2;
        bonus2 = 2;
    }
    else if (buttonLabel == "x3") {
        sum = total*3;
        bonus3 = 3;
    }
    else
        sum = total + Integer.parseInt(buttonLabel);
}

// check if the problem is solved
public boolean isSolved()
{
    int check=1, i=0;
    while (i<MAXNUM) {
        if (button[i].isVisible()==true)
            check = 0;
        i++;
    }
    if (check == 0)
        return false;
    else
        return true;
}

// set the button color
public void setButtonColor(int i)
{
    switch(number[i]) {
        case 1: button[i].setBackground(Color.blue);
```

```
                break;
            case 2: button[i].setBackground(Color.red);
                break;
            case 3: button[i].setBackground(Color.green);
                break;
            case 4: button[i].setBackground(Color.pink);
                break;
            case 5: button[i].setBackground(Color.orange);
                break;
            case 6: button[i].setBackground(Color.cyan);
                break;
            case 7: button[i].setBackground(Color.lightGray);
                break;
            case 8: button[i].setBackground(Color.magenta);
                break;
            case 9: button[i].setBackground(Color.white);
                break;
            case 10: button[i].setBackground(Color.yellow);
                 break;
            default: break;
        }
    }
}

// class Title
class Title extends JPanel
{
    private String title;
    private Font f;

    public Title()
    {
        f = new Font("Arial", Font.BOLD+Font.ITALIC, 35);
    }

    // draw the title string
    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.setFont(f);
        g.fillRect(130, 15, 350, 50);
        g.clearRect(135, 20, 350, 50);
        g.setColor(Color.blue);
        g.drawString("PLUS   X   PUZZLE", 150, 55);
    }
```

```
    // get Title panel size
    public Dimension getPreferredSize()
    {
        return new Dimension(600, 80);
    }
}

// class Score
class Score extends JPanel
{
    private Font f;
    public Score()
    {
        f = new Font("Century", Font.BOLD, 25);
    }

    public void paintComponent(Graphics g)
    {
        super.paintComponent(g);
        g.setFont(f);
        g.setColor(Color.pink);
        g.fillRoundRect(10, 250, 110, 110, 20, 20);
        g.setColor(Color.gray);
        g.fillRoundRect(15, 255, 100, 100, 20, 20);
        g.setColor(Color.pink);
        g.drawString("Plus", 35, 280);
        g.drawString("X", 55, 310);
        g.drawString("Puzzle", 25, 340);
    }

    // get Score panel size
    public Dimension getPreferredSize()
    {
        return new Dimension(130, 300);
    }
}
```