Unit7 – ICS4U0 – Android Interfaces

Sample Test – May 21, 2024

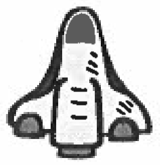
Name: Gorski

Total	%	Knowledge	Communication	Thinking	Application
(91)	%	(19)	(22)	(25)	(25)

Knowledge

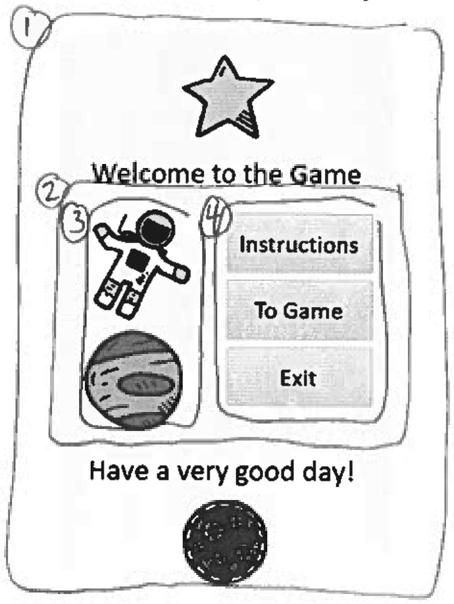
1. Identify the type of View shown in each picture. /6

Planet



Planet	(a) TextView	COUNT UP!	(c) Button	Email	(e) EditText
	(b) Image View		(d) Image View	NASA	(f) Image View

2. Outline the LinearLayouts needed around the views they enclose. (4 marks)



3. Fill in the hex codes for each colour. (5 marks) /9

Red	FF 00 00
Green	00 FF 00
Blue	00 00 FF
Yellow	FF FF 00
Cyan	00 FF FF
Magenta	FF 00 FF
Black	00 00 00
White	FF FF FF

4. Fill in this code: /4

You have 5 points `android:id="@+id/score"` change to: You have lost the game.	`TextView` Score `= (TextView) findViewById(R.id.`Score`);` Score`.setText("`You have lost the game`");`

Name
`android:id="@+id/first"`
get the text and clear the EditText.
	`EditText` first `= (EditText) findViewById(R.id.`first`);`
	`String name =` first`.getText().toString();`
	first`.setText("");`

Communication

5. Select true or false for each statement. /6

- T [✓ F] a) The attribute used to link a button with a method in java is an id. onclick
- T [✓ F] b) dp stands for dimension-independent-pixel. density
- [✓ T] F c) Android screens are laid out using XML and their actions are coded using java.
- [✓ T] F d) An onClick attribute can be added to an ImageView.
- [✓ T] F e) Android view ids can have capitals.
- T [✓ F] f) An example of a ViewGroup is a Parent. example of ViewGroup & a Parent is a Linear Layout

6. Circle the valid Android image names. Can't have capitals; normal variable naming rules /2

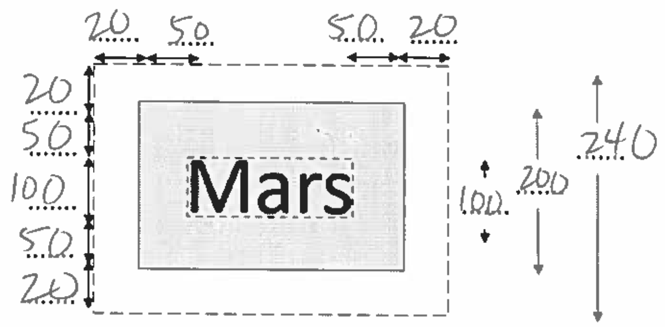
Comet [✓ comet] COMET 1comet oneComet [✓ comet1]

7. Circle the valid Android onClick values. normal variable naming rules /2

[✓ Comet] [✓ comet] [✓ COMET] 1comet [✓ oneComet] [✓ comet1]

8. Using the code, fill in the dimensions of the button. /4

```
<TextView
    android:text="Mars"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="100sp"
    android:padding="50dp"
    android:layout_margin="20dp"/>
```



9. Fill in the appropriate term that matches the description in the front column. /5

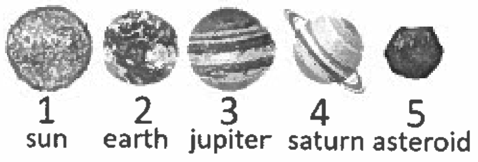
id	(a) An attribute needed if the view will be changed.
View view	(b) The parameter of the method associated with an onClick.
Linear Layout	(c) A ViewGroup with an orientation attribute.
attribute	(d) A property inside a tag that can be altered.
on Click	(e) An attribute that is needed if the View can be clicked.

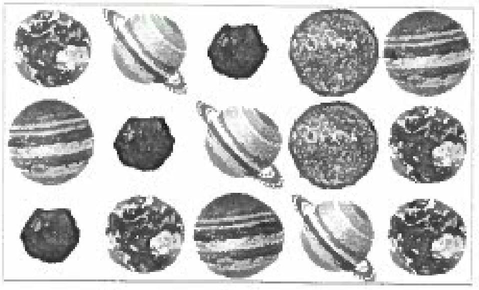
10. Define inflation and describe why it is useful in Android development. /3

① Inflation is the process of translating XML to Java.
② It occurs at the beginning of the onCreate method in the Java file.
③ It allows us to have the best of both worlds: screen set up is easier in XML; Coding actions is easier in Java. Inflation allows both to be possible.

Application

11. Using the pictures shown, fill in the code for this game of Space Swap. /6





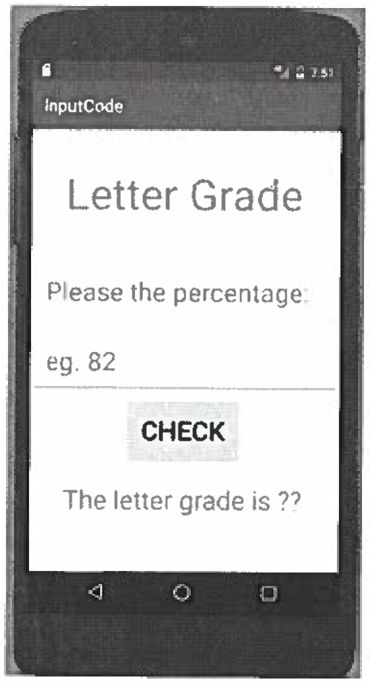
```
public class MainActivity extends AppCompatActivity {

    int space[][] = {{2, 4, 5, 1, 3},
                     {3, 5, 4, 1, 2},
                     {5, 2, 3, 4, 2}};

    int row = 3;
    int col = 5;
    ImageView pics[] = new ImageView[row * col];

protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   GridLayout g = (GridLayout) findViewById(R.id.grid);
   int m = 0;

   for (int i = 0; i < row; i++) {
      for (int j = 0; j < col; j++) {
         pics[m] = new ImageView(this);
         setpic(pics[m], m);
         pics[m].setId(m);
         g.addView(pics[m]);
         m++;
      }
   }
}
```

```
public void setpic(ImageView i, int pos) {
   int x = pos / col;
   int y = pos % col;                    match above
   int picnum = space [x][y];
   if (picnum == 1)
      i.setImageResource(R.drawable.sun);
   else if (picnum == 2)
      i.setImageResource(R.drawable.earth);
   else if (picnum == 3)
      i.setImageResource(R.drawable.jupiter);
   else if (picnum == 4)
      i.setImageResource(R.drawable.saturn);
   else
      i.setImageResource(R.drawable.asteroid);
}
```

12. The Space Swap game (from question 11) will need a redraw method.
Fill in the blanks and then complete the rest of the code that is required. /5

```
public void redraw() {
   int m = 0;
   for (int i=0 ; i < row ; i++) {
      for (int j=0 ; j<col ; j++) {
         if (space[i][j] == 1)
            pics[m].setImageResource(R.drawable.sun);
         else if (space[i][j]==2)
            pics[m].setImageResource(R.drawable.earth);
         else if (space[i][j]==3)
            pics[m].setImageResource(R.drawable.jupiter);
         else if (space[i][j]==4)
            pics[m].setImageResource(R.drawable.saturn);
         else
            pics[m].setImageResource(R.drawable.asteroid);
         m++;   ← notice
      }
   }
```

13. Fill in the blanks to create the below app. Also fill in the java for the onClick: /14


InputCode

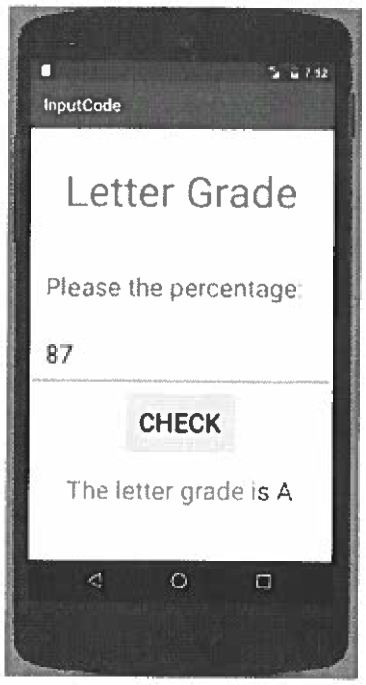
Letter Grade

Please the percentage:

eg. 82

CHECK

The letter grade is ??


InputCode

Letter Grade

Please the percentage:

87

CHECK

The letter grade is A

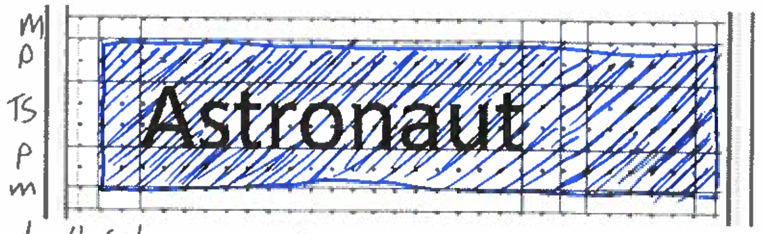
Percent	Grade
90+	A+
80-89	A
70-79	B
60-69	C
50-59	D
49-	F

```
<?xml version="1.0" encoding="utf-8"?>
<Linear Layout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout-height="match_parent"
    android:orientation="vertical">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:padding="40dp"
        android:text="Letter Grade"
        android:textSize="50sp" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="20dp"
        android:text="Please the percentage:"
        android:textSize="30sp" />
    <EditText
        android:id="@+id/input"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="eg. 82"
        android:padding="20dp"
        android:textSize="30sp" />
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:onClick="clicked"        match below
        android:padding="20dp"
        android:text="CHECK"
        android:textSize="30sp" />
    <TextView
        android:id="@+id/output"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:padding="20dp"
        android:text="The letter grade is ??"
        android:textSize="30sp" />
</Linear Layout>

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void clicked(View view) {
        EditText input = (EditText) findViewById(R.id.input);
        TextView output = (TextView) findViewById(R.id.output);
        int num = Integer.parseInt(input.getText().toString());
        String mark = "F";
        if (num >= 90)
            mark = "A+";
        else if (num >= 80)
            mark = "A";
        else if (num >= 70)
            mark = "B";
        else if (num >= 60)
            mark = "C";
        else if (num >= 50)
            mark = "D";
        output.setText("The letter grade is " + mark);
    }
} //class
```

Thinking

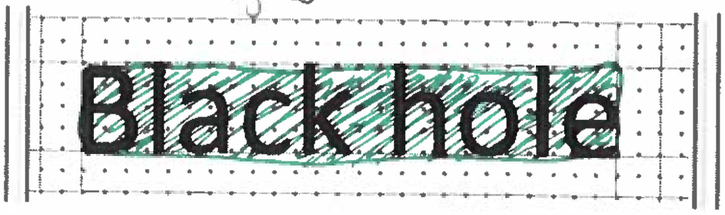
14. Fill in these views using **colour**. The width of the screen is shown. Each dot square is 10 dp. /4

(a)
```
<Button
  android:text="Astronaut"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:textSize="30sp" fix
  android:layout_margin="10sp"
  android:background="#0000FF"
  android:padding="20dp"/>
```



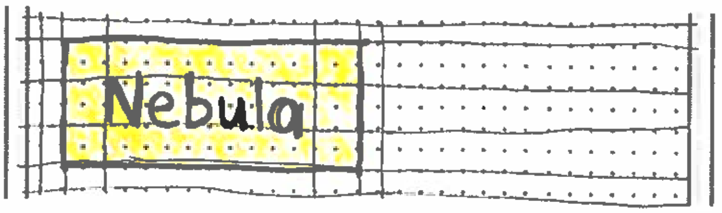
✱ colour padding, don't colour margins

(b)
```
<Button
  android:text="Blackhole"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:layout_margin="20sp"
  android:textSize="40sp" fix
  android:background="#00FF00"/>
```

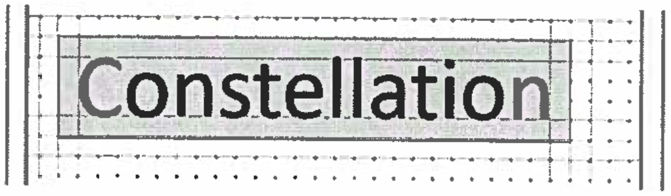


15. Draw this view using **colour**. The width of the screen is shown. Each dot square is 10 dp. /5

```
<Button
  android:text="Nebula"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textSize="20sp"
  android:background="#FFFF00"
  android:padding="20dp"
  android:layout_margin="10dp"/>
```



16. Fill in the code for this view. The width of the screen is shown. Each dot square is 10 dp. /5



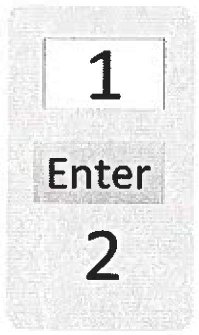
```
<Button
  android:text="Constellation"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textSize="30sp"
  android:background="#CCCCCC"
  android:padding="10dp"
  android:layout_margin="10dp"/>
```

17. Circle **and correct** five errors in this code. actually 6 ☺ /5

```
<~~b~~Button
  android:text="Reverse"
  android:layout_width="wrap_content"
  android:layout_~~width~~="wrap_content" height
  android:textSize="40sp"
  android:background="#FFFF00"~~"~~
  android:padding="10dp"
  android:onClick="reverse"
  android:id="@+id/backwards"
/>
```

18. A number of test cases were run on this app and the results of each test are shown in the table below. What is the code in the onClick method named `enter`? (The editText's id is `input`; the textView's is `output`)

/6



Test Case	Output
2	6
-10	90
4	20
-5	20
-0.5	-0.25
0	0
-1	0

```
1  public void enter(View view){
2  EditText input =(EditText) findViewById (R.id.input);
2  TextView output=(TextView)findViewById (R.id.output);
3  double n= Double.parseDouble (input.getText().toString());
4  double ans = n*n +n;
5  output.setText(""+ans);
   }
```

sort to help
find pattern

-10	90
-5	20
-1	0
-0.5	-0.25
0	0
2	6
4	20

$n*n+n$
$4*4+4=20$

Steps
1. method line
2. handle inflation, find widgets
3. get data from textfield (EditText)
4. do stuff
5. output result