# Northern Michigan University (Marquette Co, MI)

CS345-01-25W Android Programming (Andrew A. Poe) Name: ______________________________
Midterm Examination (Exam 1) Page 1/2 Thursday 27 February 2025 4:00 P.M. EST

Time: 50 minutes.

1. Write the following method in Kotlin:

fun RingAroundTheArray (A:Array<String>):String { ... }

A is an array of string such the length of each string in the array is the same as the length of all the other strings. This method returns a string generated by starting at the upper left corner of A and running clockwise around the array of string. For example, if the array were {ABCDE , FGHIJ, KLMNO }, written as a table it would look like:

```
ABCDE
FGHIJ
KLMNO
```

and the returned string would be ABCDEJONMLKF

```
var S = ""
if (A.size > 0)
 if (A.size==1) S = A[0]
 else {
  var cols = A[0].length
  var rows = A.size
  for (i in 0..cols-2) S += A[0][i]
  for (i in 0..rows-2) S += A[i][cols-1]
  for (i in cols-1 downTo 1) S += A[rows-1][i]
  for (i in rows-1 downto 1) S += A[i][0]
 }
return S
```

2. Android uses an Activity Lifecycle to handle apps with more than one screen layout or function. iOS doesn't do this and so the programmer switches between different views to accomplish the same thing, all of which are active but only one of which is visible. What might be an advantage to the way Android does it, with activity lifecycles?

**The big issue is the once an activity ends, it is completely destroyed and is no longer using resources at all. Even when an activity is inactive, it stops and isn't consuming CPU time at all.**

3. My app has a button myButt, and text views myString and myCount linked in the XML to fields called button1, TV1, and TV2 respectively. Write that portion of onCreate that will link the Kotlin widgets to the XML definitions and defines the button so that clicking on it will do the following: it will count the number of words in myString and place that count in myCount. For example, if myWord contains "Spring Break Spring Break", "4" will be placed in myCount. For the sake of this problem, a word is a maximal contiguous group of characters not containing a space. In other words, words are separated by one or more spaces, and there may be leading and trailing spaces, too.

```
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    myButt = findViewById (R.id.button1)
    myString = findViewById (R.id.TV1)
    myCount = findViewById (R.id.TV2)

    myButt!!.setOnClickListener {

          var prev = ' '
          var wct = 0
          var s = myString!!.text
          for (ch in s) {
             if (prev==' ' && ch != ' ') wct++
             prev = ch
          myCount!!.text = ""+wct
    }
}
```

4. My app has two activities: the main activity and AActivity. Each activity has a button called myButt. Whenever I click myButt in the main activity, AActivity should start. Whenever I click myButt in AActivity, the main activity should start. Every time I click a button, the other activity should start.

Write the code within these two activities and the main activity that will make this work. Explain why this is perhaps not the best way to switch between the two activities.

```
AActivity:
fun newIntent (packageContext:Context?):Intent? {

 return Intent (packageContext,AActivity::class.java)
}

onCreate { ...

 myButt.setOnClickListener {
  var i:Intent? = MainActivity.newIntent(this)
  startActivity (i)
 }
}

MainActivity:
fun newIntent (packageContext:Context?):Intent? {

 return Intent (packageContext,MainActivity::class.java)
}

onCreate { ...

 myButt.setOnClickListener {
  var i:Intent? = MainActivity.newIntent(this)
  startActivity (i)
 }
}
```

**If you keep calling activities without returning from any of them, you end up wasting resources as multiple copies of each activity keep piling up. Better to have main call A and have A return to main.**