Polytechnic Tutoring Center

Midterm 1 REVIEW – CS1133, Fall 2020

Disclaimer: This mock exam is only for practice. It was made by tutors in the Polytechnic Tutoring Center and is not representative of the actual exam given by the Academic Department.

Question 1

```
clear; clc;
PARKS = given; % you are given PARKS

minDist = 100;
maxDist = 800;

% circle around the park 3 times
nLaps = 3;
% maximum time spent at the park
maxTimeHours = 2;
hoursToMinutes = 60; % conversion factor to go from hours to minutes
maxTime = maxTimeHours/hoursToMinutes;
% you will need to divide the maximum time Jerry wants to spend at the
% park by the number of times Jerry wants to walk around the park to get
% the maximum amount of time that can be spent on a single lap around
% the park.
maxOneLap = maxTime/nLaps;

% Jerry wants there to be a concessions stand
snackS = true;

% create a variable for each column index
colDist = 1;
colTime = 2;
colSnack = 3;

% there are conditions for distance, time, and snacks
FarEnough = PARKS(:,colDist)>minDist;
TooFar = PARKS(:,colDist)>maxDist;
RightDistance = FarEnough & ~TooFar; % an exclusive interval
Time = PARKS(:,colTime)<=minDist;
% note that no more than 2 minutes includes 2 minutes, so we use less
% than or equal to as opposed to just less than
Snacks = PARKS(:,colSnack)==snackS;
Condition = RightDistance & Time & Snacks;

% we use and because all of the conditions must be true
GoodParks = find(Condition);
FarAwayParks = find(TooFar);
```

Question 2

```
clear; clc; format short;

nVal = input('Enter an odd number: ');

OddVec = 1:nVal; % Creating vector of odd numbers
nRows = nVal+1; % Row dimension for new matrix
nCols = nVal;

theMiddle = round(nVal/2); % Finding middle column
RowIdx = 2:nRows;

MTX = zeros(nRows,nCols);
MTX(1,:) = OddVec; % First row equal to vector

MTX(RowIdx,theMiddle) = OddVec; % Middle column equal to vector
```

Question 3

```
clear; clc;
nTrials = 1e8; % Some large number of trials
diceFaces = 6; % faces of a a die
coinSides = 2; % sides of a coin

Rolls = randi([1,diceFaces],1,nTrials); % random rolling of the die
FlipsOne = randi([1,coinSides],1,nTrials); % random flipping of coin 1
FlipsTwo = randi([1,coinSides],1,nTrials); % random flipping of coin 2

aTwo = 2;
aFive = 5;
aHeads = 1; % You can use either 1 or 2 to represent heads or tails

twoOrFive = Rolls == aTwo | Rolls == aFive;
HeadsOne = FlipsOne == aHeads;
HeadsTwo = FlipsTwo ~= aHeads;
% twoOrFive and Heads are both logical vectors

Condition = twoOrFive & HeadsOne & HeadsTwo; % a logical vector; note that you
% must use & not |

nTimesTrue = nnz(Condition); % the number of times both conditions were true
Probability = nTimesTrue/nTrials;
```

Extra Practice

```
clear; clc; format short g;

nRows = 3; nCols = 5;
MTX0 = zeros(nRows,nCols);
MTX0(:) = randperm(nRows*nCols)';

% Creating copies of the original array so that we have the correct size for
```

```
% arranging. You can also create an array of zeros.
MTX1 = MTX0;
MTX2 =  MTX0;

nCols2 = nCols/2;
endFirst = floor(nCols2);
startSecond = ceil(nCols2) + 1;

MTX1(:,startSecond:end) = MTX0(:,endFirst:-1:1)
MTX2(:,1:endFirst) = MTX0(:,end:-1:startSecond)
```