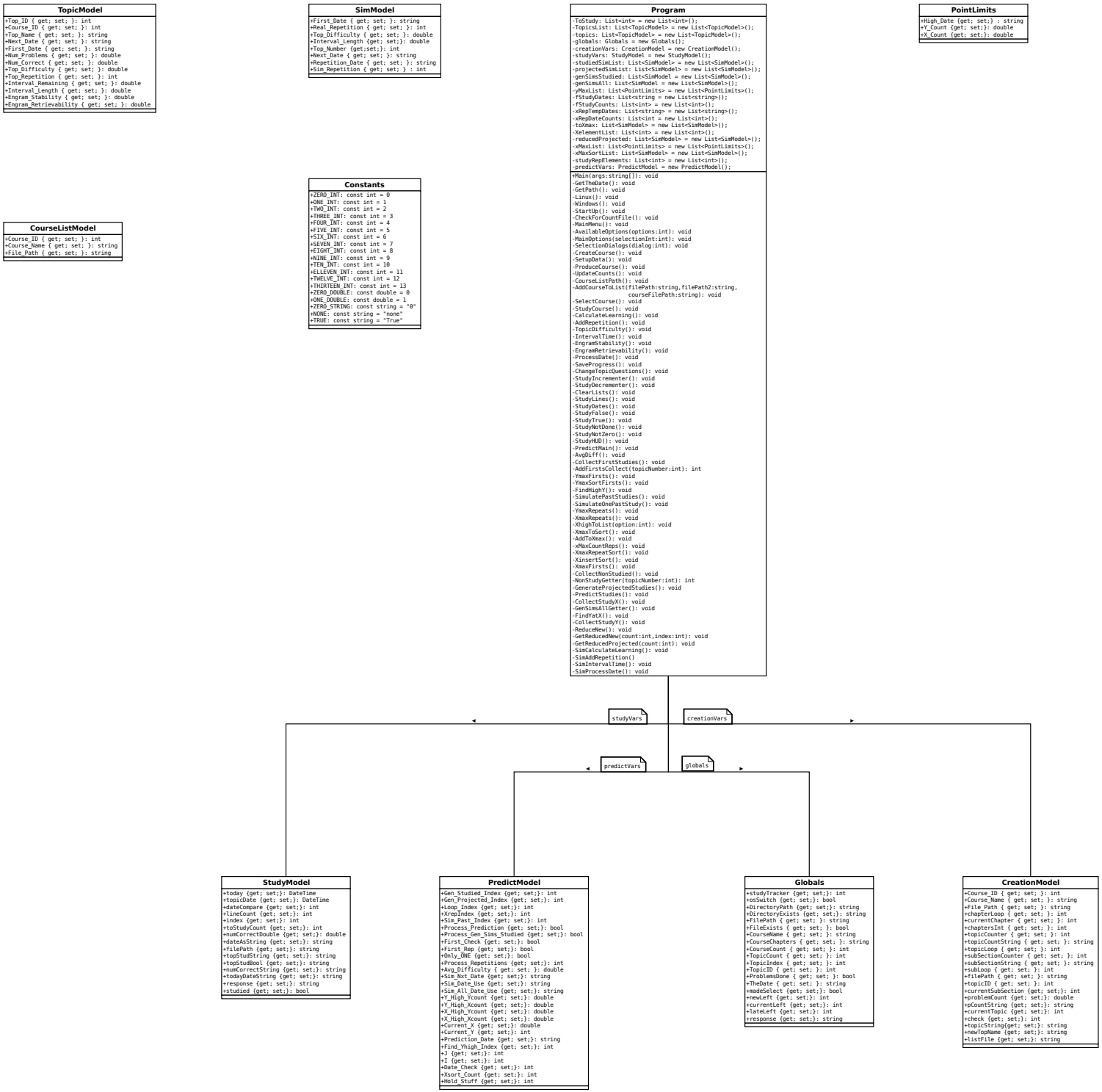
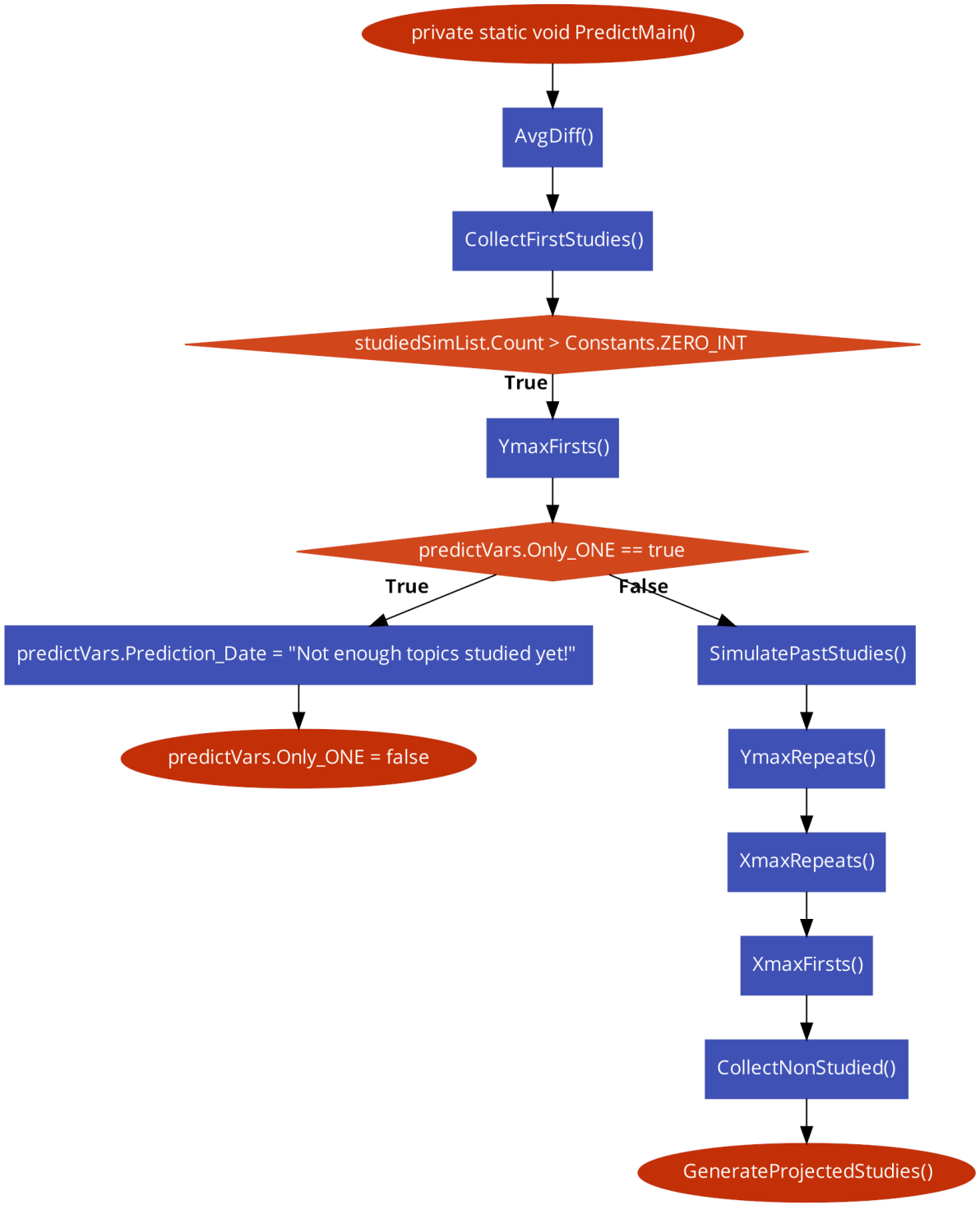
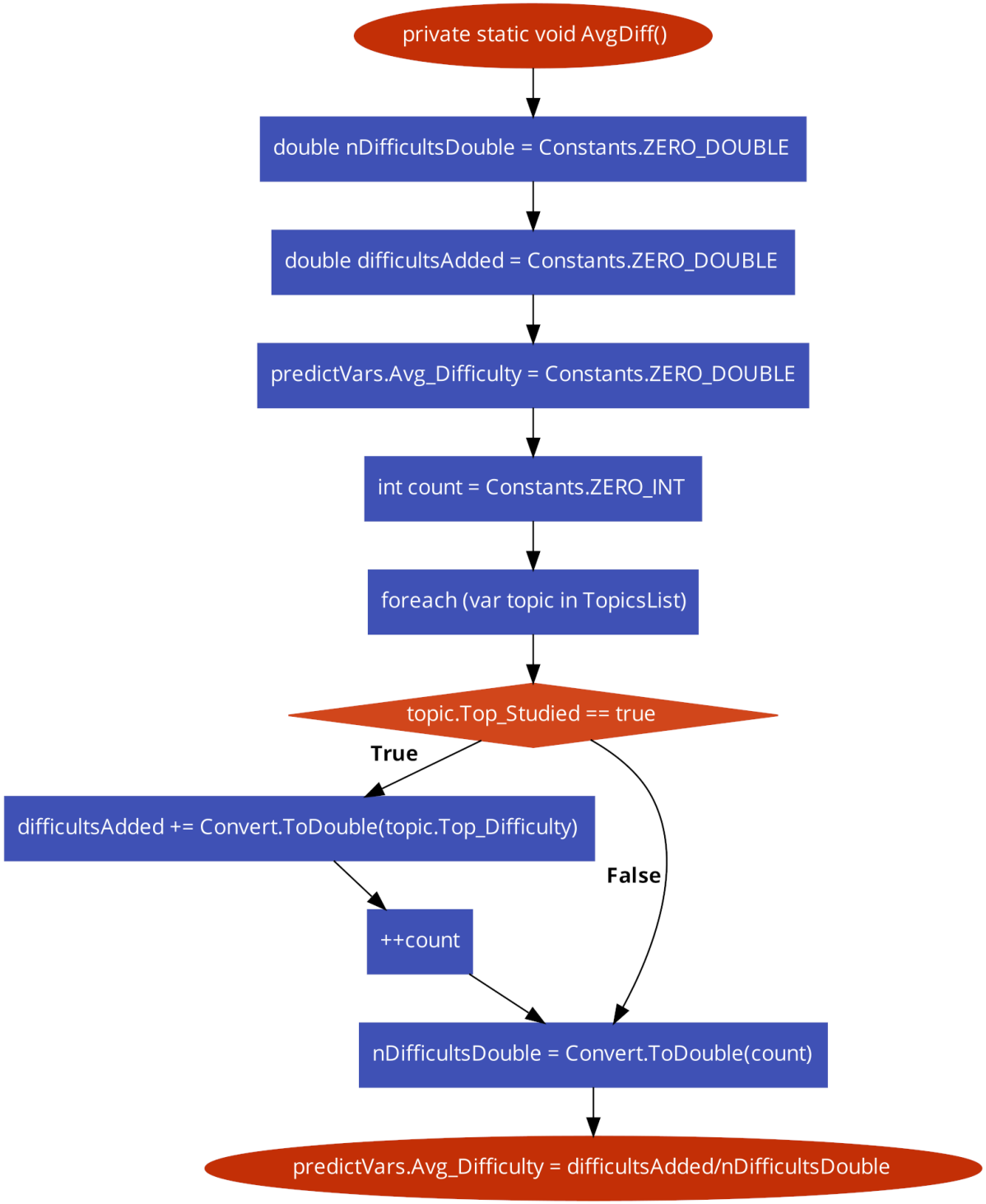
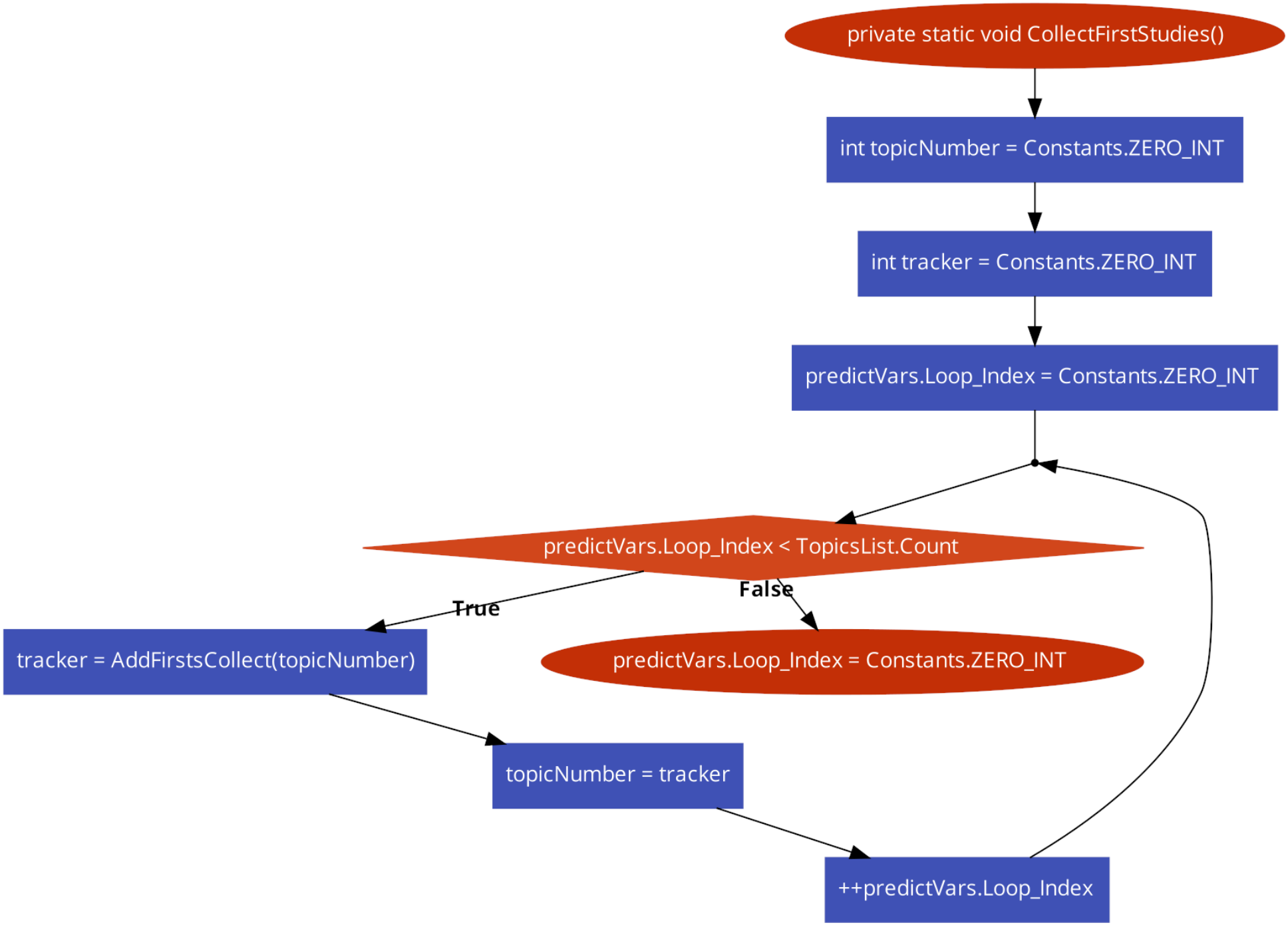
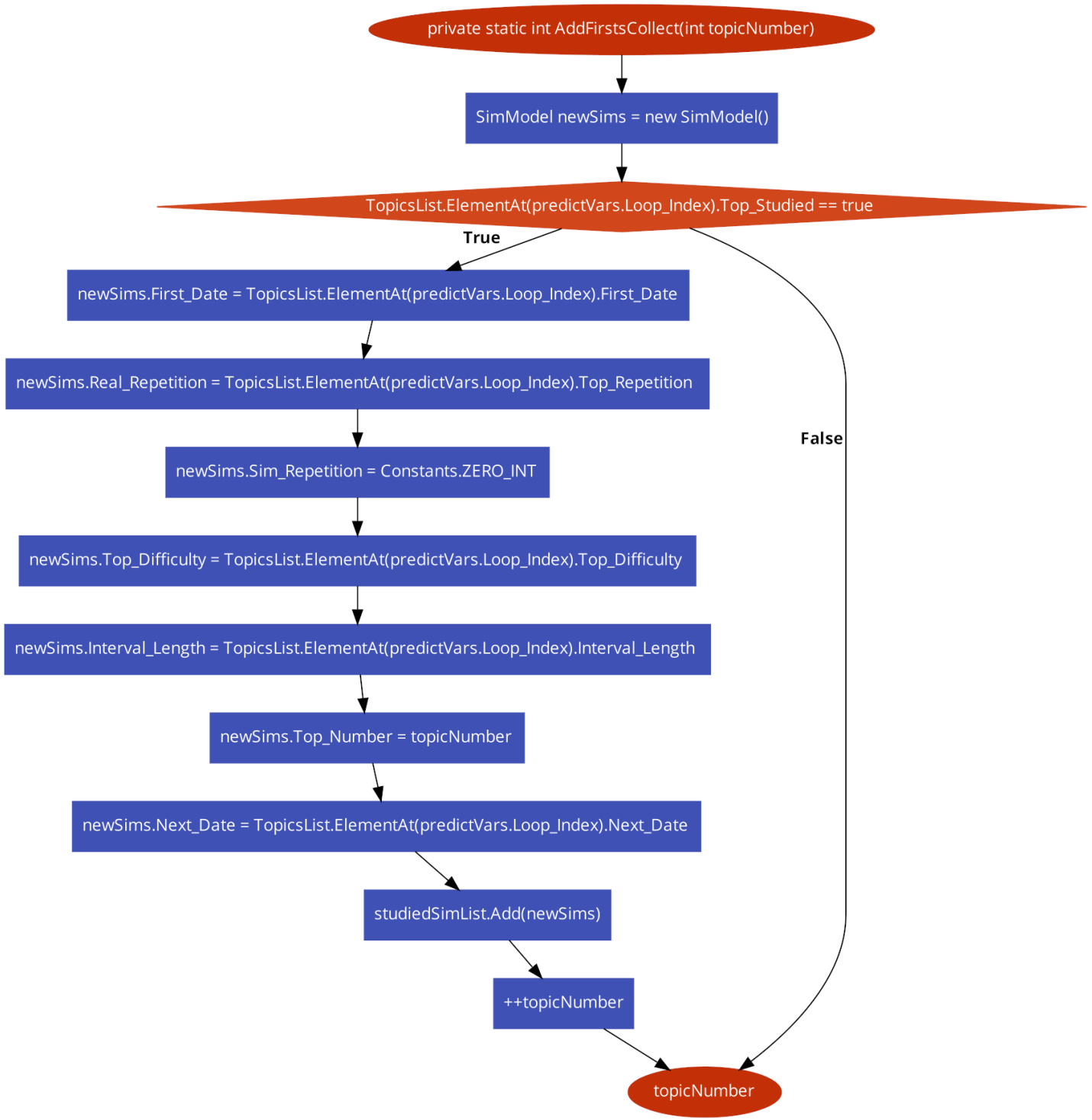
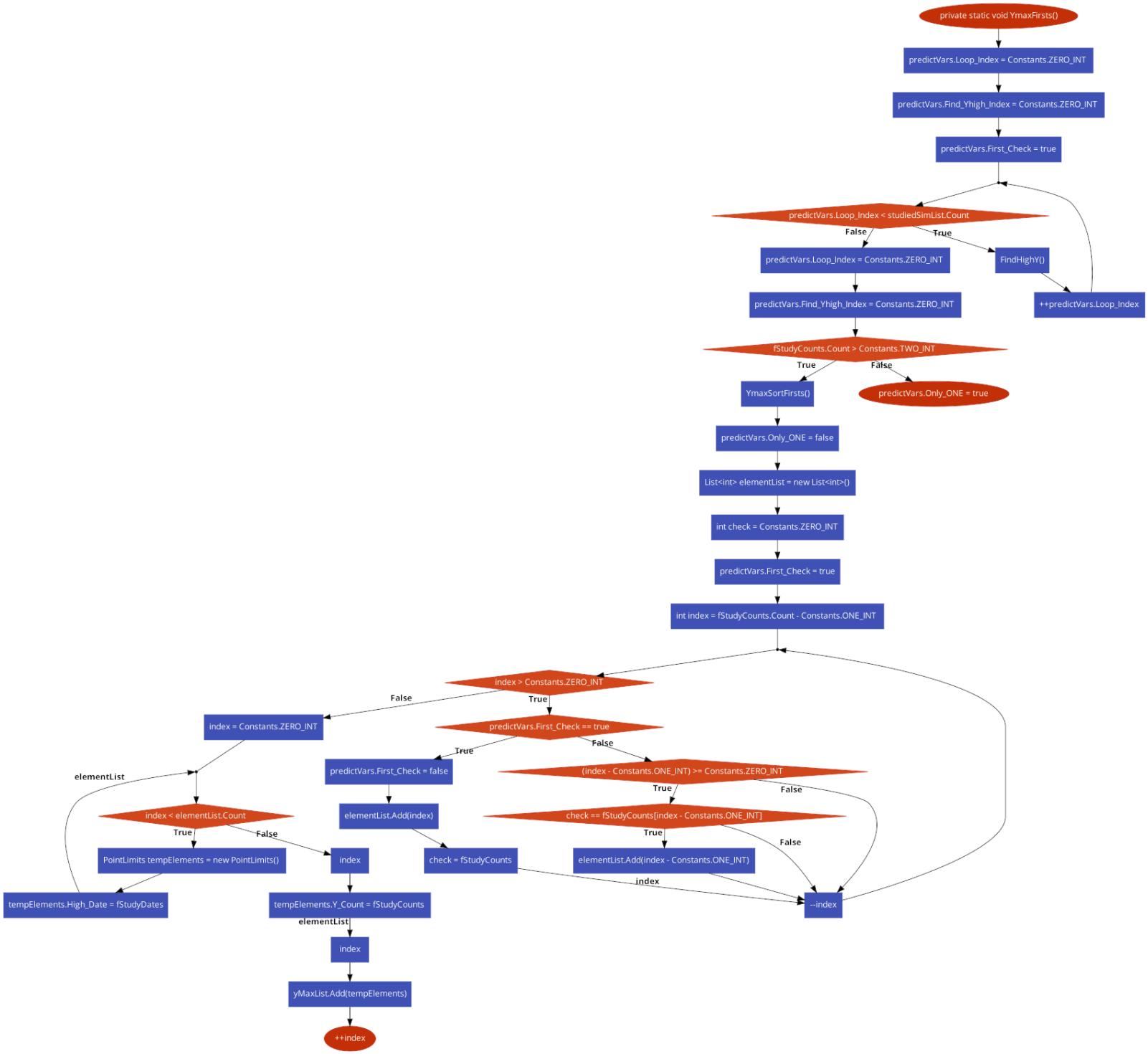

TopicModel
+Top_ID { get; set; }: int
+Course_ID { get; set; }: int
+Top_Name { get; set; }: string
+Next_Date { get; set; }: string
+First_Date { get; set; }: string
+Num_Problems { get; set; }: double
+Num_Correct { get; set; }: double
+Top_Difficulty { get; set; }: double
+Top_Repetition { get; set; }: int
+Interval_Remaining { get; set; }: double
+Interval_Length { get; set; }: double
+Engram_Stability { get; set; }: double
+Engram_Retrievability { get; set; }: double
SimModel
+First_Date { get; set; }: string
+Real_Repetition { get; set; }: int
+Top_Difficulty { get; set; }: double
+Interval_Length {get; set;}: double
+Top_Number {get;set;}: int
+Next_Date { get; set; }: string
+Repetition_Date { get; set; }: string
+Sim_Repetition { get; set; } : int
Program
-ToStudy: List<int> = new List<int>();
-TopicsList: List<TopicModel> = new List<TopicModel>();
-topics: List<TopicModel> = new List<TopicModel>();
-globals: Globals = new Globals();
-creationVars: CreationModel = new CreationModel();
-studyVars: StudyModel = new StudyModel();
-studiedSimList: List<SimModel> = new List<SimModel>();
-projectedSimList: List<SimModel> = new List<SimModel>();
-genSimsStudied: List<SimModel = new List<SimModel>();
-genSimsAll: List<SimModel = new List<SimModel>();
-yMaxList: List<PointLimits> = new List<PointLimits>();
-fStudyDates: List<string = new List<string>();
-fStudyCounts: List<int> = new List<int>();
-xRepTempDates: List<string> = new List<string>();
-xRepDateCounts: List<int = new List<int>();
-toXmax: List<SimModel> = new List<SimModel>();
-XelementList: List<int> = new List<int>();
-reducedProjected: List<SimModel> = new List<SimModel>();
-xMaxList: List<PointLimits> = new List<PointLimits>();
-xMaxSortList: List<SimModel> = new List<SimModel>();
-studyRepElements: List<int> = new List<int>();
-predictVars: PredictModel = new PredictModel();
+Main(args:string[]): void
-GetTheDate(): void
-GetPath(): void
-Linux(): void
-Windows(): void
-StartUp(): void
-CheckForCountFile(): void
-MainMenu(): void
-AvailableOptions(options:int): void
-MainOptions(selectionInt:int): void
-SelectionDialogs(dialog:int): void
-CreateCourse(): void
-SetupData(): void
-ProduceCourse(): void
-UpdateCounts(): void
-CourseListPath(): void
-AddCourseToList(filePath:string,filePath2:string,
courseFilePath:string): void
-SelectCourse(): void
-StudyCourse(): void
-CalculateLearning(): void
-AddRepetition(): void
-TopicDifficulty(): void
-IntervalTime(): void
-EngramStability(): void
-EngramRetrievability(): void
-ProcessDate(): void
-SaveProgress(): void
-ChangeTopicQuestions(): void
-StudyIncrementer(): void
-StudyDecrementer(): void
-ClearLists(): void
-StudyLines(): void
-StudyDates(): void
-StudyFalse(): void
-StudyTrue(): void
-StudyNotDone(): void
-StudyNotZero(): void
-StudyHUD(): void
-PredictMain(): void
-AvgDiff(): void
-CollectFirstStudies(): void
-AddFirstsCollect(topicNumber:int): int
-YmaxFirsts(): void
-YmaxSortFirsts(): void
-FindHighY(): void
-SimulatePastStudies(): void
-SimulateOnePastStudy(): void
-YmaxRepeats(): void
-XmaxRepeats(): void
-XhighToList(option:int): void
-XmaxToSort(): void
-AddToXmax(): void
-xMaxCountReps(): void
-XmaxRepeatSort(): void
-XinsertSort(): void
-XmaxFirsts(): void
-CollectNonStudied(): void
-NonStudyGetter(topicNumber:int): int
-GenerateProjectedStudies(): void
-PredictStudies(): void
-CollectStudyX(): void
-GenSimsAllGetter(): void
-FindYatX(): void
-CollectStudyY(): void
-ReduceNew(): void
-GetReducedNew(count:int,index:int): void
-GetReducedProjected(count:int): void
-SimCalculateLearning(): void
-SimAddRepetition()
-SimIntervalTime(): void
-SimProcessDate(): void
PointLimits
+High_Date {get; set;} : string
+Y_Count {get; set;}: double
+X_Count {get; set;}: double
Constants
+ZERO_INT: const int = 0
+ONE_INT: const int = 1
+TWO_INT: const int = 2
+THREE_INT: const int = 3
+FOUR_INT: const int = 4
+FIVE_INT: const int = 5
+SIX_INT: const int = 6
+SEVEN_INT: const int = 7
+EIGHT_INT: const int = 8
+NINE_INT: const int = 9
+TEN_INT: const int = 10
+ELLEVEN_INT: const int = 11
+TWELVE_INT: const int = 12
+THIRTEEN_INT: const int = 13
+ZERO_DOUBLE: const double = 0
+ONE_DOUBLE: const double = 1
+ZERO_STRING: const string = "0"
+NONE: const string = "none"
+TRUE: const string = "True"
CourseListModel
+Course_ID { get; set; }: int
+Course_Name { get; set; }: string
+File_Path { get; set; }: string
studyVars
creationVars
predictVars
globals
StudyModel
+today {get; set;}: DateTime
+topicDate {get; set;}: DateTime
+dateCompare {get; set;}: int
+timeCount {get; set;}: int
+index {get; set;}: int
+toStudyCount {get; set;}: int
+numCorrectDouble {get; set;}: double
+dateAsString {get; set;}: string
+filePath {get; set;}: string
+topStudString {get; set;}: string
+topStudBool {get; set;}: string
+numCorrectString {get; set;}: string
+todayDateString {get; set;}: string
+response {get; set;}: string
+studied {get; set;}: bool
PredictModel
+Gen_Studied_Index {get; set;}: int
+Gen_Projected_Index {get; set;}: int
+Loop_Index {get; set;}: int
+XrepIndex {get; set;}: int
+Sim_Past_Index {get; set;}: int
+Process_Prediction {get; set;}: bool
+Process_Gen_Sims_Studied {get; set;}: bool
+First_Check {get; set;}: bool
+First_Rep {get; set;}: bool
+Only_ONE {get; set;}: bool
+Process_Repetitions {get; set;}: int
+Avg_Difficulty { get; set; }: double
+Sim_Nxt_Date {get; set;}: string
+Sim_Date_Use {get; set;}: string
+Sim_All_Date_Use {get; set;}: string
+Y_High_Ycount {get; set;}: double
+Y_High_Xcount {get; set;}: double
+X_High_Ycount {get; set;}: double
+X_High_Xcount {get; set;}: double
+Current_X {get; set;}: double
+Current_Y {get; set;}: int
+Prediction_Date {get; set;}: string
+Find_Yhigh_Index {get; set;}: int
+J {get; set;}: int
+I {get; set;}: int
+Date_Check {get; set;}: int
+Xsort_Count {get; set;}: int
+Hold_Stuff {get; set;}: int
Globals
+studyTracker {get; set;}: int
+osSwitch {get; set;}: bool
+DirectoryPath {get; set;}: string
+DirectoryExists {get; set;}: string
+FilePath { get; set; }: string
+FileExists { get; set; }: bool
+CourseName { get; set; }: string
+CourseChapters { get; set; }: string
+CourseCount { get; set; }: int
+TopicCount { get; set; }: int
+TopicIndex { get; set; }: int
+TopicID { get; set; }: int
+ProblemsDone { get; set; }: bool
+TheDate { get; set; }: string
+madeSelect {get; set;}: bool
+newLeft {get; set;}: int
+currentLeft {get; set;}: int
+lateLeft {get; set;}: int
+response {get; set;}: string
CreationModel
+Course_ID { get; set; }: int
+Course_Name { get; set; }: string
+File_Path { get; set; }: string
+chapterLoop { get; set; }: int
+currentChapter { get; set; }: int
+chaptersInt { get; set; }: int
+topicCounter { get; set; }: int
+topicCountString { get; set; }: string
+topicLoop { get; set; }: int
+subSectionCounter { get; set; }: int
+subSectionString { get; set; }: string
+subLoop { get; set; }: int
+filePath { get; set; }: string
+topicID { get; set; }: int
+currentSubSection {get; set;}: int
+problemCount {get; set;}: double
+pCountString {get; set;}: string
+currentTopic {get; set;}: int
+check {get; set;}: int
+topicString{get; set;}: string
+newTopName {get; set;}: string
+listFile {get; set;}: string

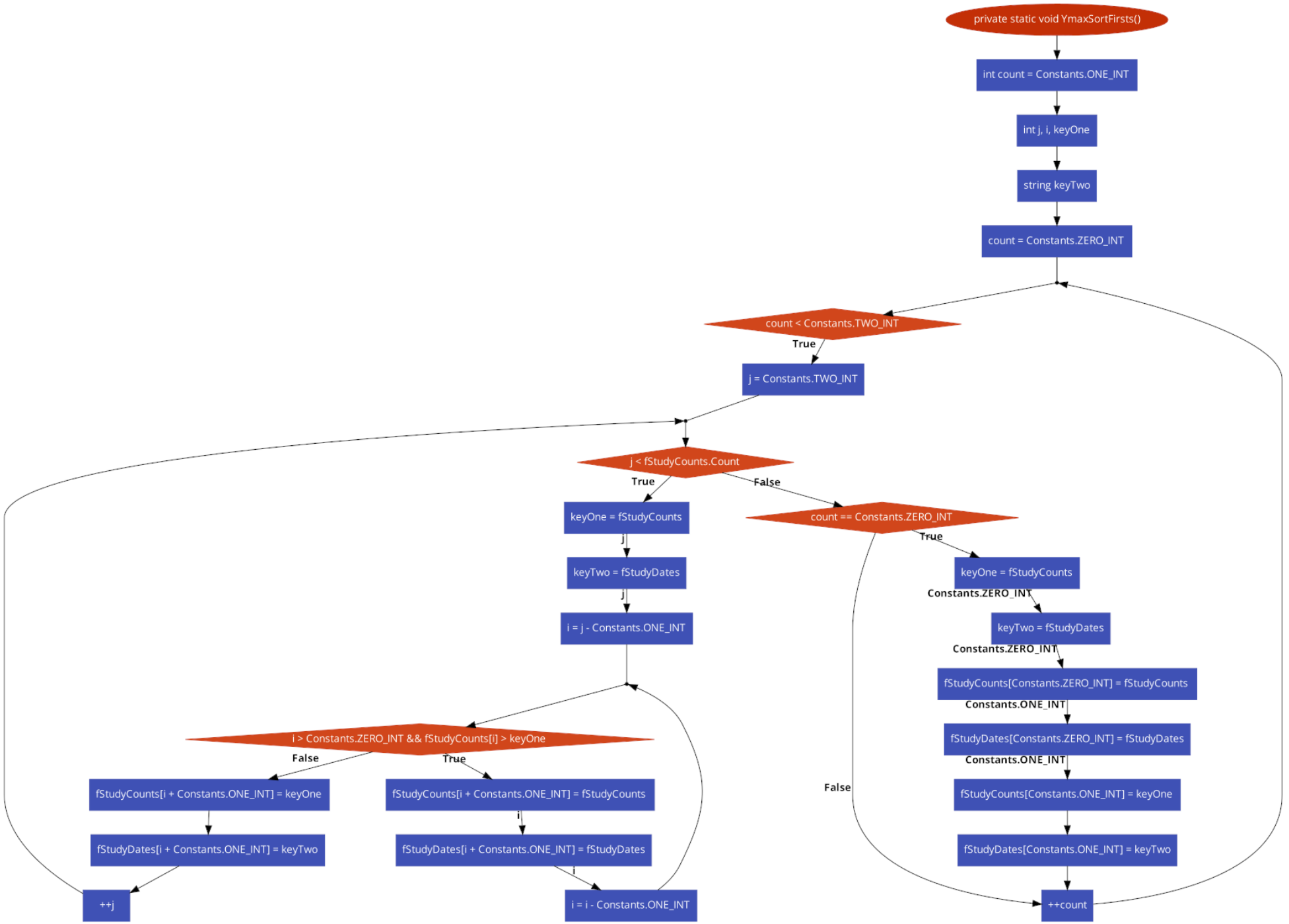

private static void PredictMain()
AvgDiff()
CollectFirstStudies()
studiedSimList.Count > Constants.ZERO_INT
True
YmaxFirsts()
predictVars.Only_ONE == true
True
False
predictVars.Prediction_Date = "Not enough topics studied yet!"
predictVars.Only_ONE = false
SimulatePastStudies()
YmaxRepeats()
XmaxRepeats()
XmaxFirsts()
CollectNonStudied()
GenerateProjectedStudies()

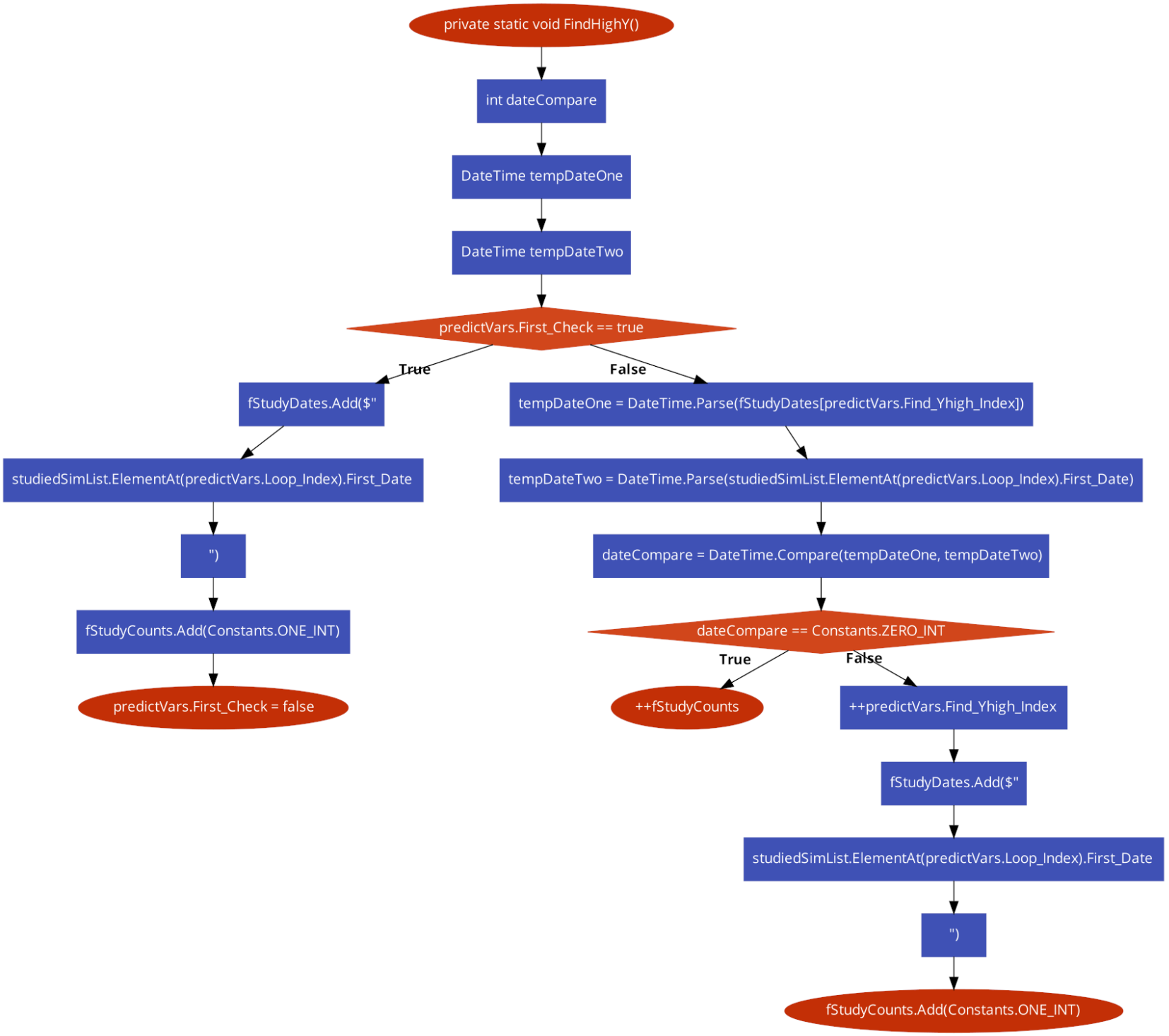

private static void AvgDiff()
double nDifficultsDouble = Constants.ZERO_DOUBLE
double difficultsAdded = Constants.ZERO_DOUBLE
predictVars.Avg_Difficulty = Constants.ZERO_DOUBLE
int count = Constants.ZERO_INT
foreach (var topic in TopicsList)
topic.Top_Studied == true
True
False
difficultsAdded += Convert.ToDouble(topic.Top_Difficulty)
++count
nDifficultsDouble = Convert.ToDouble(count)
predictVars.Avg_Difficulty = difficultsAdded/nDifficultsDouble

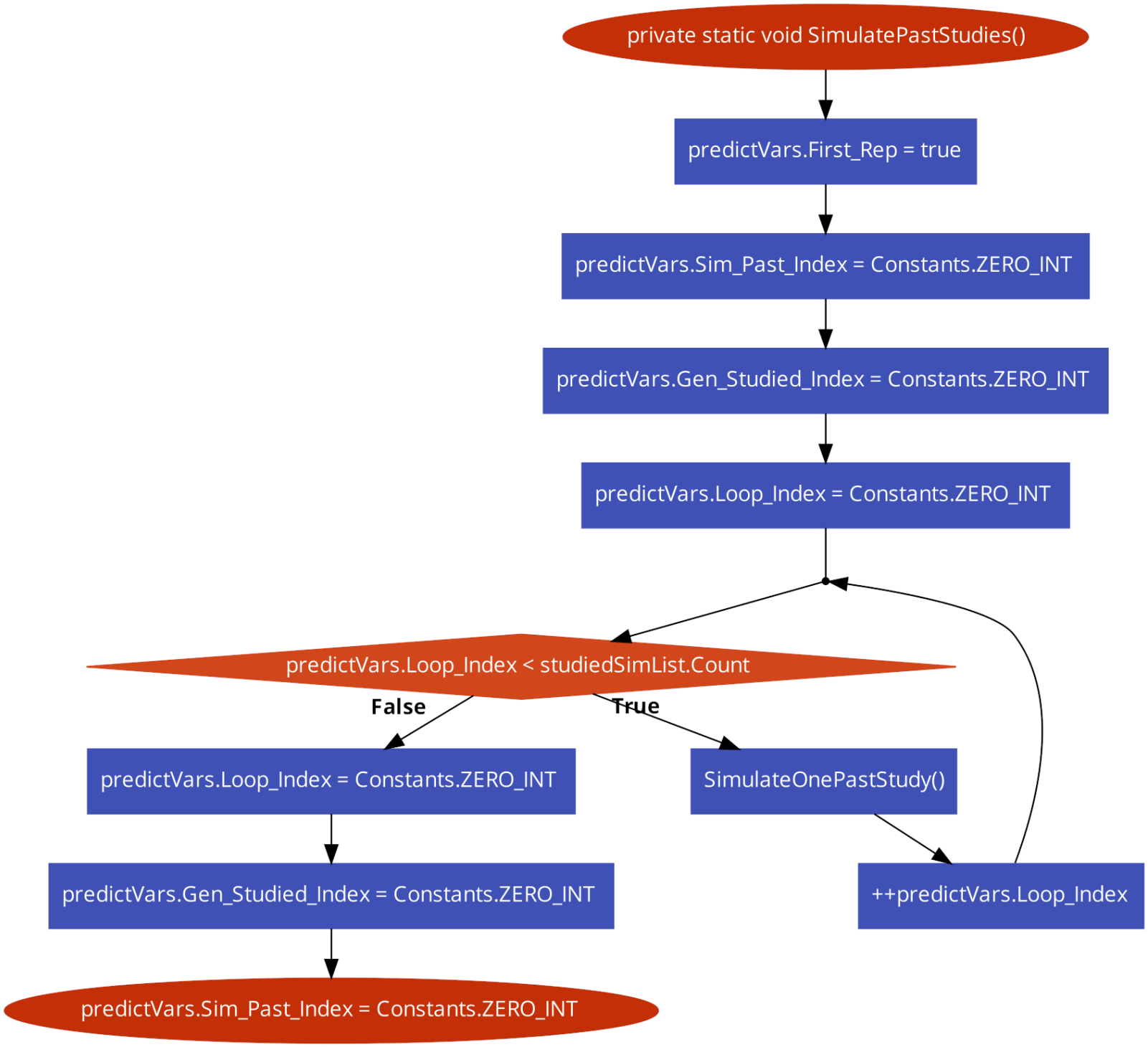

private static void CollectFirstStudies()
int topicNumber = Constants.ZERO_INT
int tracker = Constants.ZERO_INT
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.Loop_Index < TopicsList.Count
True
False
tracker = AddFirstsCollect(topicNumber)
predictVars.Loop_Index = Constants.ZERO_INT
topicNumber = tracker
++predictVars.Loop_Index

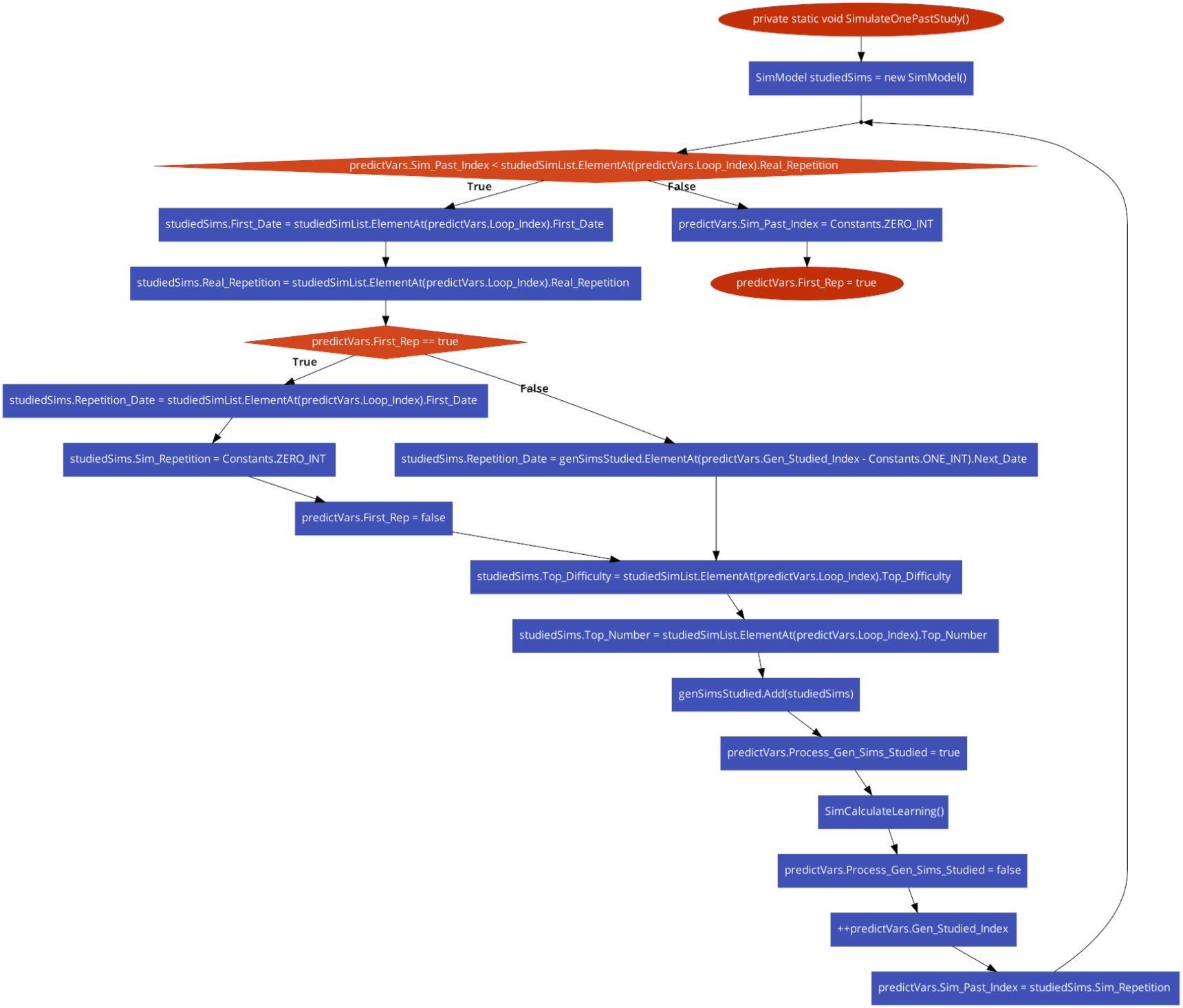

private static int AddFirstsCollect(int topicNumber)
SimModel newSims = new SimModel()
TopicsList.ElementAt(predictVars.Loop_Index).Top_Studied == true
True
False
newSims.First_Date = TopicsList.ElementAt(predictVars.Loop_Index).First_Date
newSims.Real_Repetition = TopicsList.ElementAt(predictVars.Loop_Index).Top_Repetition
newSims.Sim_Repetition = Constants.ZERO_INT
newSims.Top_Difficulty = TopicsList.ElementAt(predictVars.Loop_Index).Top_Difficulty
newSims.Interval_Length = TopicsList.ElementAt(predictVars.Loop_Index).Interval_Length
newSims.Top_Number = topicNumber
newSims.Next_Date = TopicsList.ElementAt(predictVars.Loop_Index).Next_Date
studiedSimList.Add(newSims)
++topicNumber
topicNumber

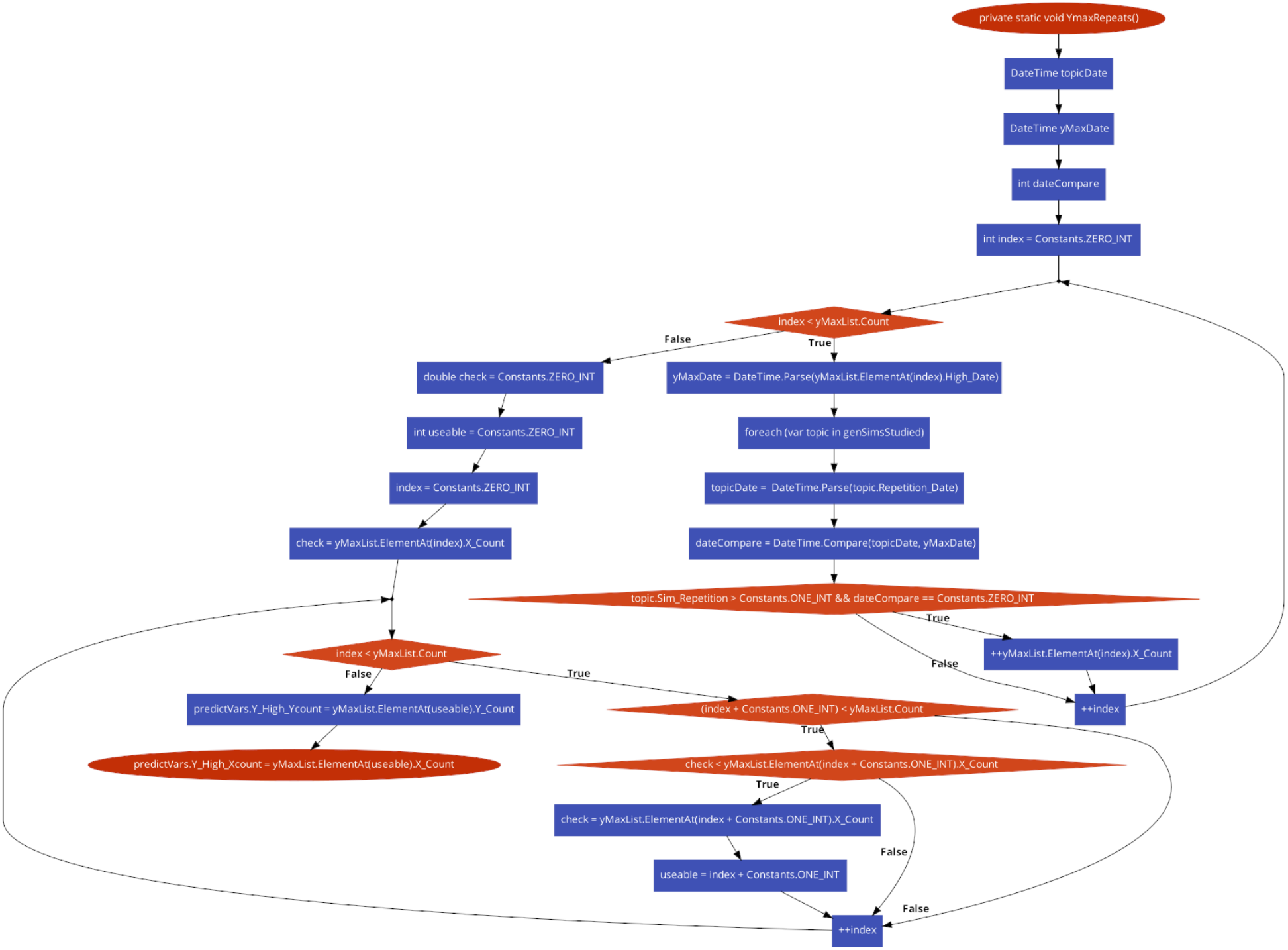

private static void YmaxFirsts()
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.Find_Yhigh_Index = Constants.ZERO_INT
predictVars.First_Check = true
predictVars.Loop_Index < studiedSimList.Count
False
True
predictVars.Loop_Index = Constants.ZERO_INT
FindHighY()
predictVars.Find_Yhigh_Index = Constants.ZERO_INT
++predictVars.Loop_Index
fStudyCounts.Count > Constants.TWO_INT
True
False
YmaxSortFirsts()
predictVars.Only_ONE = true
predictVars.Only_ONE = false
List<int> elementList = new List<int>()
int check = Constants.ZERO_INT
predictVars.First_Check = true
int index = fStudyCounts.Count - Constants.ONE_INT
index > Constants.ZERO_INT
False
True
index = Constants.ZERO_INT
predictVars.First_Check == true
True
False
predictVars.First_Check = false
(index - Constants.ONE_INT) >= Constants.ZERO_INT
True
False
elementList
index < elementList.Count
True
False
elementList.Add(index)
check == fStudyCounts[index - Constants.ONE_INT]
True
False
PointLimits tempElements = new PointLimits()
index
check = fStudyCounts
elementList.Add(index - Constants.ONE_INT)
index
tempElements.High_Date = fStudyDates
tempElements.Y_Count = fStudyCounts
--index
elementList
index
yMaxList.Add(tempElements)
++index

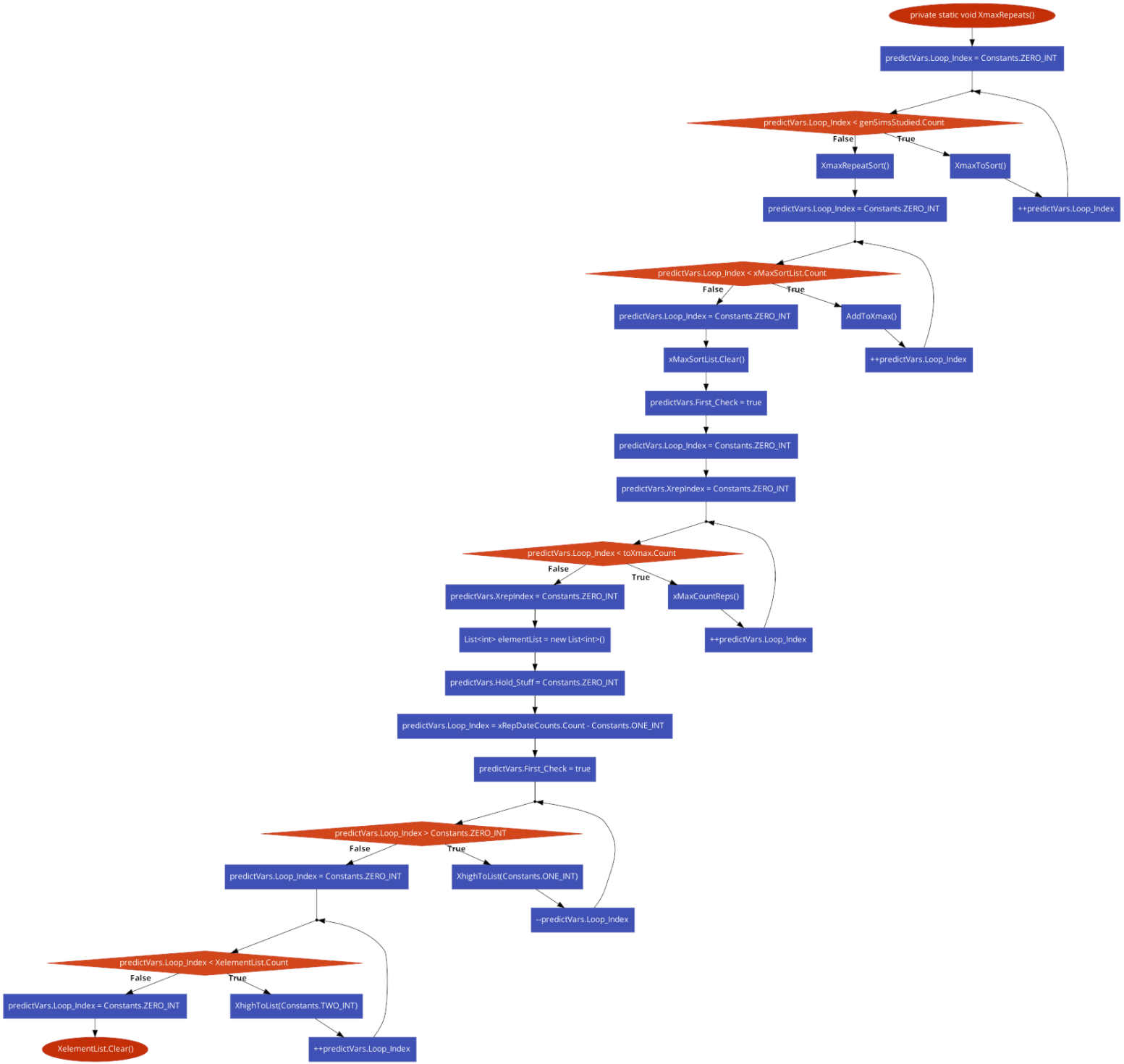

private static void YmaxSortFirsts()
int count = Constants.ONE_INT
int j, i, keyOne
string keyTwo
count = Constants.ZERO_INT
count < Constants.TWO_INT
True
j = Constants.TWO_INT
j < fStudyCounts.Count
True
False
keyOne = fStudyCounts
J
keyTwo = fStudyDates
J
i = j - Constants.ONE_INT
i > Constants.ZERO_INT && fStudyCounts[i] > keyOne
False
True
fStudyCounts[i + Constants.ONE_INT] = keyOne
fStudyDates[i + Constants.ONE_INT] = keyTwo
++j
fStudyCounts[i + Constants.ONE_INT] = fStudyCounts
i
fStudyDates[i + Constants.ONE_INT] = fStudyDates
i
i = i - Constants.ONE_INT
count == Constants.ZERO_INT
True
False
keyOne = fStudyCounts
Constants.ZERO_INT
keyTwo = fStudyDates
Constants.ZERO_INT
fStudyCounts[Constants.ZERO_INT] = fStudyCounts
Constants.ONE_INT
fStudyDates[Constants.ZERO_INT] = fStudyDates
Constants.ONE_INT
fStudyCounts[Constants.ONE_INT] = keyOne
fStudyDates[Constants.ONE_INT] = keyTwo
++count

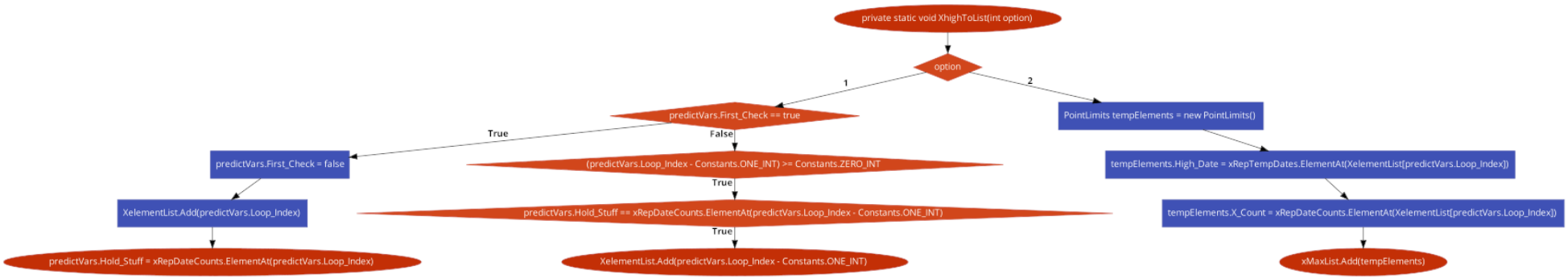

private static void FindHighY()
int dateCompare
DateTime tempDateOne
DateTime tempDateTwo
predictVars.First_Check == true
True
False
fStudyDates.Add($"
studiedSimList.ElementAt(predictVars.Loop_Index).First_Date
")
fStudyCounts.Add(Constants.ONE_INT)
predictVars.First_Check = false
tempDateOne = DateTime.Parse(fStudyDates[predictVars.Find_Yhigh_Index])
tempDateTwo = DateTime.Parse(studiedSimList.ElementAt(predictVars.Loop_Index).First_Date)
dateCompare = DateTime.Compare(tempDateOne, tempDateTwo)
dateCompare == Constants.ZERO_INT
True
False
++fStudyCounts
++predictVars.Find_Yhigh_Index
fStudyDates.Add($"
studiedSimList.ElementAt(predictVars.Loop_Index).First_Date
")
fStudyCounts.Add(Constants.ONE_INT)


private static void SimulatePastStudies()
predictVars.First_Rep = true
predictVars.Sim_Past_Index = Constants.ZERO_INT
predictVars.Gen_Studied_Index = Constants.ZERO_INT
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.Loop_Index < studiedSimList.Count
False
True
predictVars.Loop_Index = Constants.ZERO_INT
SimulateOnePastStudy()
predictVars.Gen_Studied_Index = Constants.ZERO_INT
++predictVars.Loop_Index
predictVars.Sim_Past_Index = Constants.ZERO_INT


private static void SimulateOnePastStudy()
SimModel studiedSims = new SimModel()
predictVars.Sim_Past_Index < studiedSimList.ElementAt(predictVars.Loop_Index).Real_Repetition
True
False
studiedSims.First_Date = studiedSimList.ElementAt(predictVars.Loop_Index).First_Date
predictVars.Sim_Past_Index = Constants.ZERO_INT
studiedSims.Real_Repetition = studiedSimList.ElementAt(predictVars.Loop_Index).Real_Repetition
predictVars.First_Rep = true
predictVars.First_Rep == true
True
False
studiedSims.Repetition_Date = studiedSimList.ElementAt(predictVars.Loop_Index).First_Date
studiedSims.Sim_Repetition = Constants.ZERO_INT
studiedSims.Repetition_Date = genSimsStudied.ElementAt(predictVars.Gen_Studied_Index - Constants.ONE_INT).Next_Date
predictVars.First_Rep = false
studiedSims.Top_Difficulty = studiedSimList.ElementAt(predictVars.Loop_Index).Top_Difficulty
studiedSims.Top_Number = studiedSimList.ElementAt(predictVars.Loop_Index).Top_Number
genSimsStudied.Add(studiedSims)
predictVars.Process_Gen_Sims_Studied = true
SimCalculateLearning()
predictVars.Process_Gen_Sims_Studied = false
++predictVars.Gen_Studied_Index
predictVars.Sim_Past_Index = studiedSims.Sim_Repetition

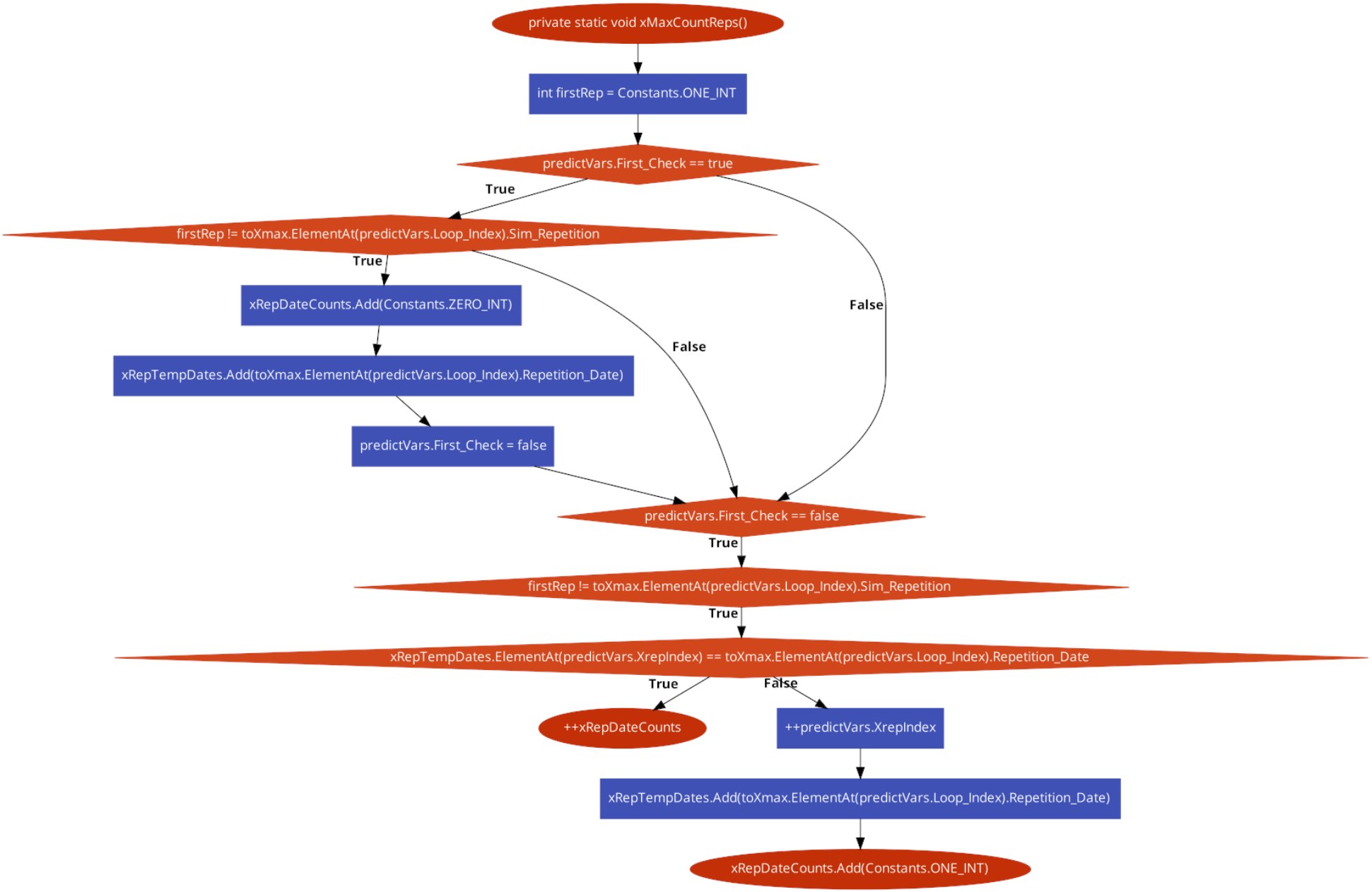

private static void YmaxRepeats()
DateTime topicDate
DateTime yMaxDate
int dateCompare
int index = Constants.ZERO_INT
index < yMaxList.Count
False
True
double check = Constants.ZERO_INT
int useable = Constants.ZERO_INT
index = Constants.ZERO_INT
check = yMaxList.ElementAt(index).X_Count
yMaxDate = DateTime.Parse(yMaxList.ElementAt(index).High_Date)
foreach (var topic in genSimsStudied)
topicDate = DateTime.Parse(topic.Repetition_Date)
dateCompare = DateTime.Compare(topicDate, yMaxDate)
topic.Sim_Repetition > Constants.ONE_INT && dateCompare == Constants.ZERO_INT
True
False
++yMaxList.ElementAt(index).X_Count
++index
index < yMaxList.Count
False
True
predictVars.Y_High_Ycount = yMaxList.ElementAt(useable).Y_Count
predictVars.Y_High_Xcount = yMaxList.ElementAt(useable).X_Count
(index + Constants.ONE_INT) < yMaxList.Count
True
check < yMaxList.ElementAt(index + Constants.ONE_INT).X_Count
True
check = yMaxList.ElementAt(index + Constants.ONE_INT).X_Count
useable = index + Constants.ONE_INT
False
False
++index

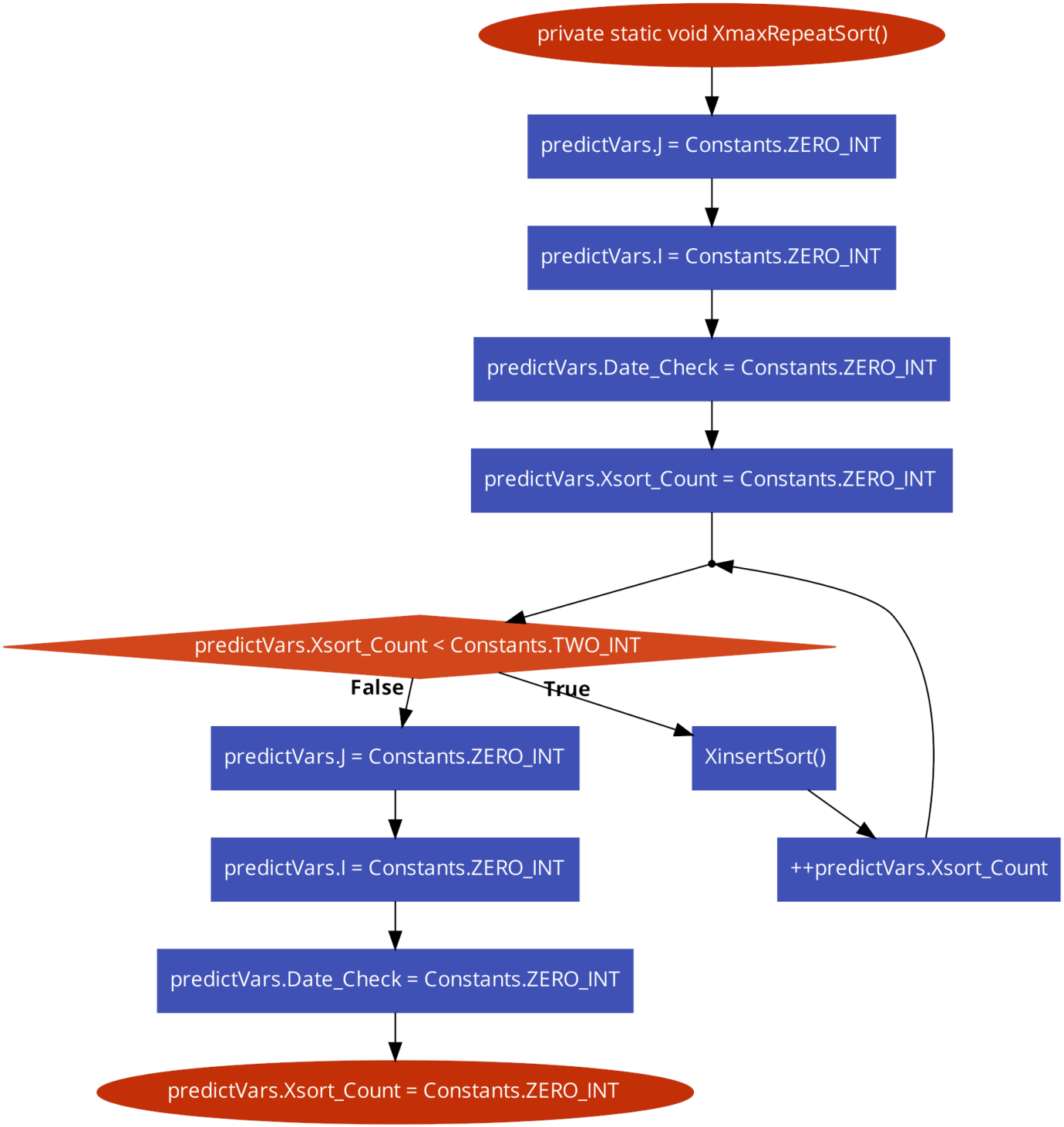

private static void XmaxRepeats()
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.Loop_Index < genSimsStudied.Count
False
True
XmaxRepeatSort()
XmaxToSort()
predictVars.Loop_Index = Constants.ZERO_INT
++predictVars.Loop_Index
predictVars.Loop_Index < xMaxSortList.Count
False
True
predictVars.Loop_Index = Constants.ZERO_INT
AddToXmax()
xMaxSortList.Clear()
++predictVars.Loop_Index
predictVars.First_Check = true
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.XrepIndex = Constants.ZERO_INT
predictVars.Loop_Index < toXmax.Count
False
True
predictVars.XrepIndex = Constants.ZERO_INT
xMaxCountReps()
List<int> elementList = new List<int>()
++predictVars.Loop_Index
predictVars.Hold_Stuff = Constants.ZERO_INT
predictVars.Loop_Index = xRepDateCounts.Count - Constants.ONE_INT
predictVars.First_Check = true
predictVars.Loop_Index > Constants.ZERO_INT
False
True
predictVars.Loop_Index = Constants.ZERO_INT
XhighToList(Constants.ONE_INT)
--predictVars.Loop_Index
predictVars.Loop_Index < XelementList.Count
False
True
predictVars.Loop_Index = Constants.ZERO_INT
XhighToList(Constants.TWO_INT)
XelementList.Clear()
++predictVars.Loop_Index

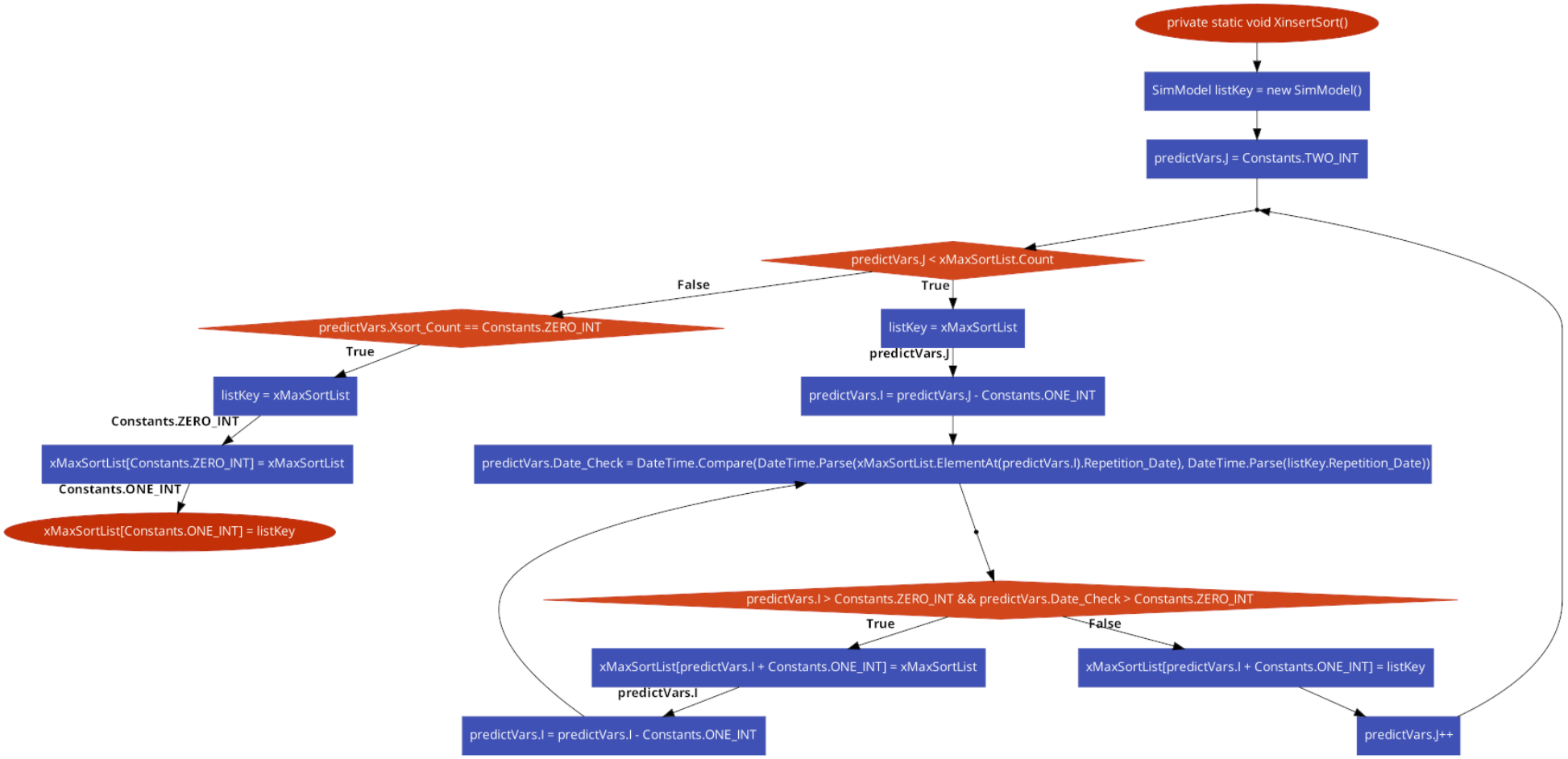

private static void XhighToList(int option)
option
1
2
predictVars.First_Check == true
PointLimits tempElements = new PointLimits()
True
False
predictVars.First_Check = false
(predictVars.Loop_Index - Constants.ONE_INT) >= Constants.ZERO_INT
tempElements.High_Date = xRepTempDates.ElementAt(XelementList[predictVars.Loop_Index])
True
XelementList.Add(predictVars.Loop_Index)
predictVars.Hold_Stuff == xRepDateCounts.ElementAt(predictVars.Loop_Index - Constants.ONE_INT)
tempElements.X_Count = xRepDateCounts.ElementAt(XelementList[predictVars.Loop_Index])
True
predictVars.Hold_Stuff = xRepDateCounts.ElementAt(predictVars.Loop_Index)
XelementList.Add(predictVars.Loop_Index - Constants.ONE_INT)
xMaxList.Add(tempElements)

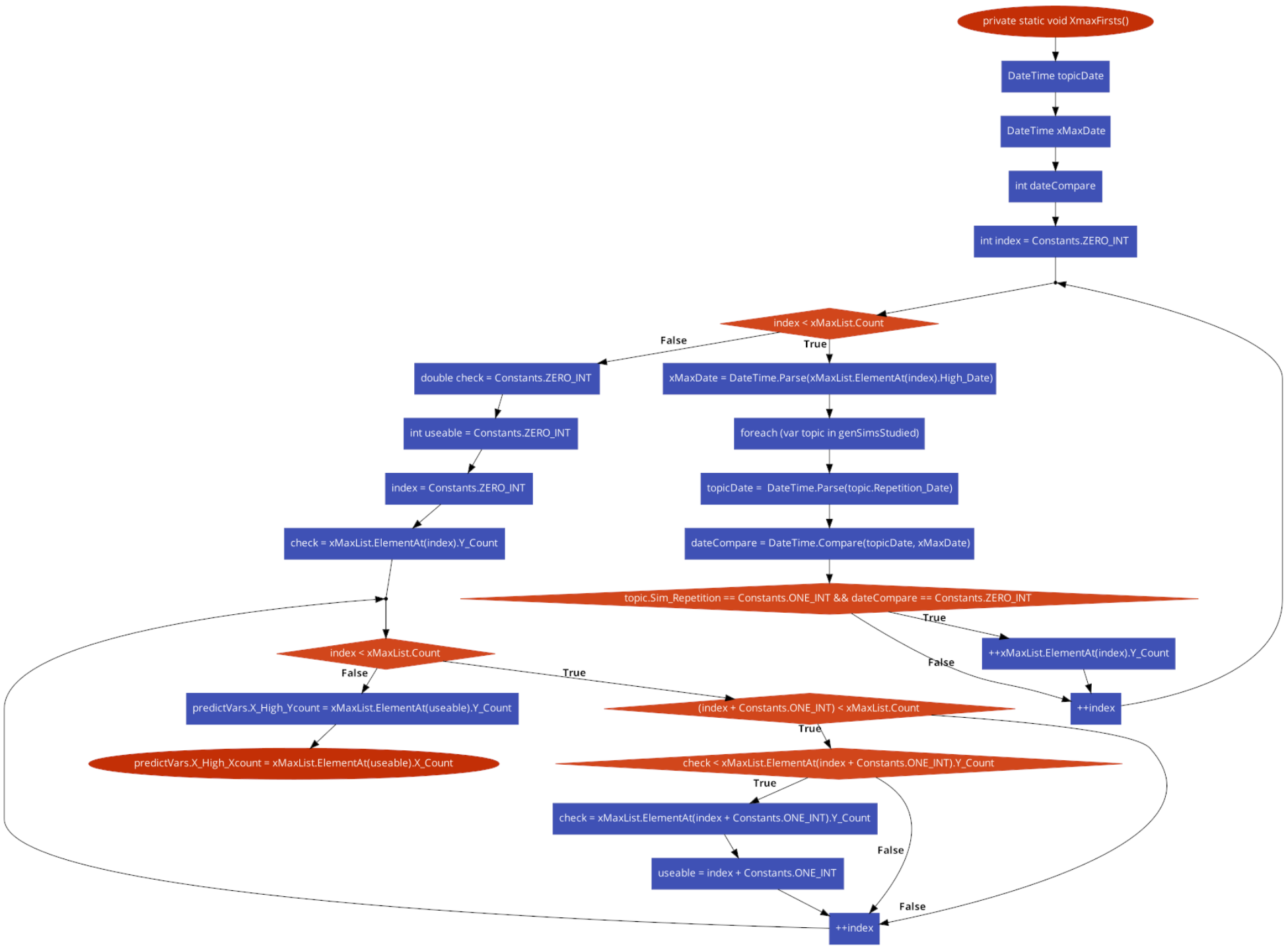

private static void XmaxToSort()
SimModel sim = new SimModel()
sim.First_Date = genSimsStudied.ElementAt(predictVars.Loop_Index).First_Date
sim.Real_Repetition = genSimsStudied.ElementAt(predictVars.Loop_Index).Real_Repetition
sim.Top_Difficulty = genSimsStudied.ElementAt(predictVars.Loop_Index).Top_Difficulty
sim.Interval_Length = genSimsStudied.ElementAt(predictVars.Loop_Index).Interval_Length
sim.Top_Number = genSimsStudied.ElementAt(predictVars.Loop_Index).Top_Number
sim.Next_Date = genSimsStudied.ElementAt(predictVars.Loop_Index).Next_Date
sim.Repetition_Date = genSimsStudied.ElementAt(predictVars.Loop_Index).Repetition_Date
sim.Sim_Repetition = genSimsStudied.ElementAt(predictVars.Loop_Index).Sim_Repetition
xMaxSortList.Add(sim)

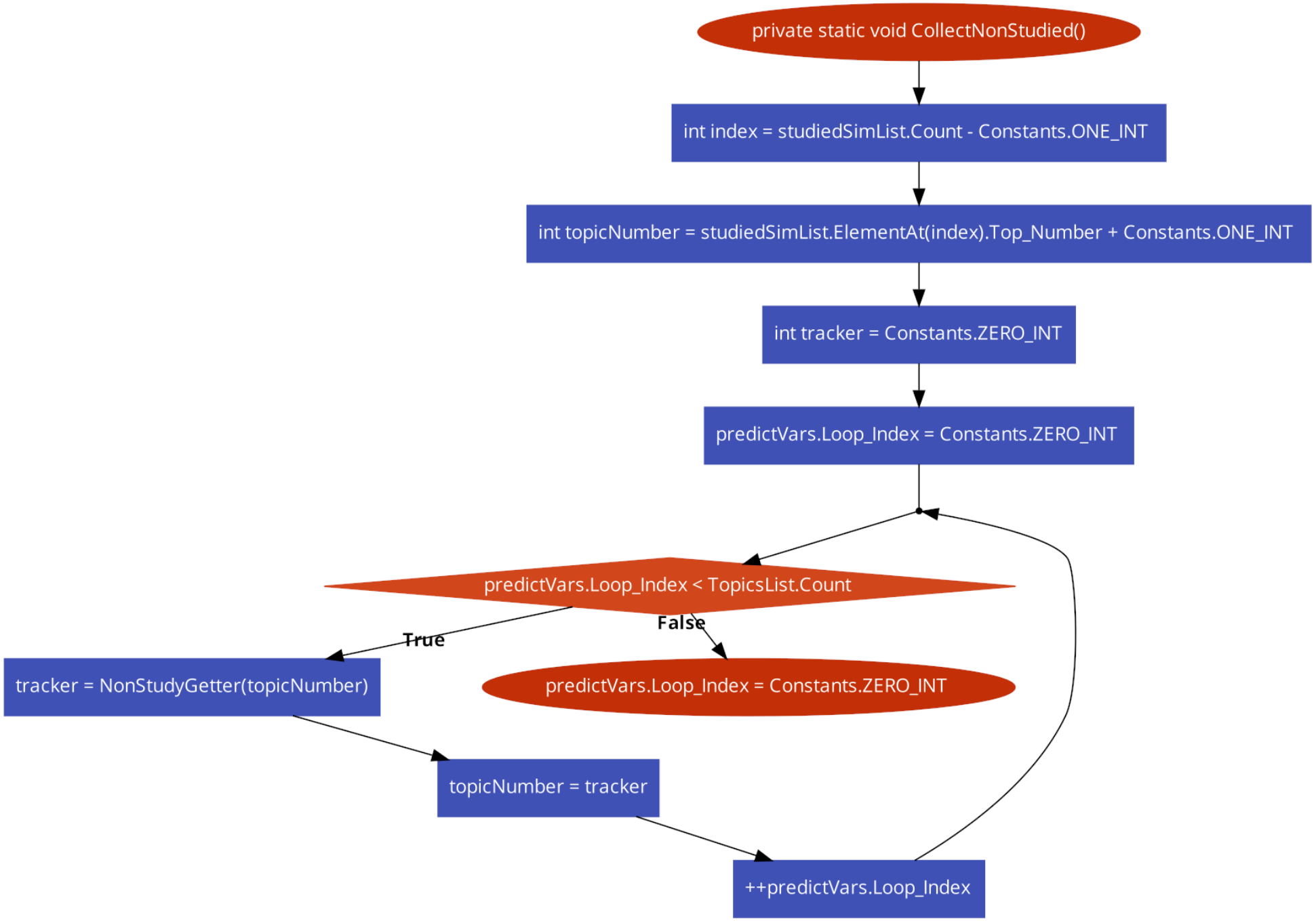

private static void AddToXmax()
SimModel sim = new SimModel()
sim.First_Date = xMaxSortList.ElementAt(predictVars.Loop_Index).First_Date
sim.Real_Repetition = xMaxSortList.ElementAt(predictVars.Loop_Index).Real_Repetition
sim.Top_Difficulty = xMaxSortList.ElementAt(predictVars.Loop_Index).Top_Difficulty
sim.Interval_Length = xMaxSortList.ElementAt(predictVars.Loop_Index).Interval_Length
sim.Top_Number = xMaxSortList.ElementAt(predictVars.Loop_Index).Top_Number
sim.Next_Date = xMaxSortList.ElementAt(predictVars.Loop_Index).Next_Date
sim.Repetition_Date = xMaxSortList.ElementAt(predictVars.Loop_Index).Repetition_Date
sim.Sim_Repetition = xMaxSortList.ElementAt(predictVars.Loop_Index).Sim_Repetition
toXmax.Add(sim)

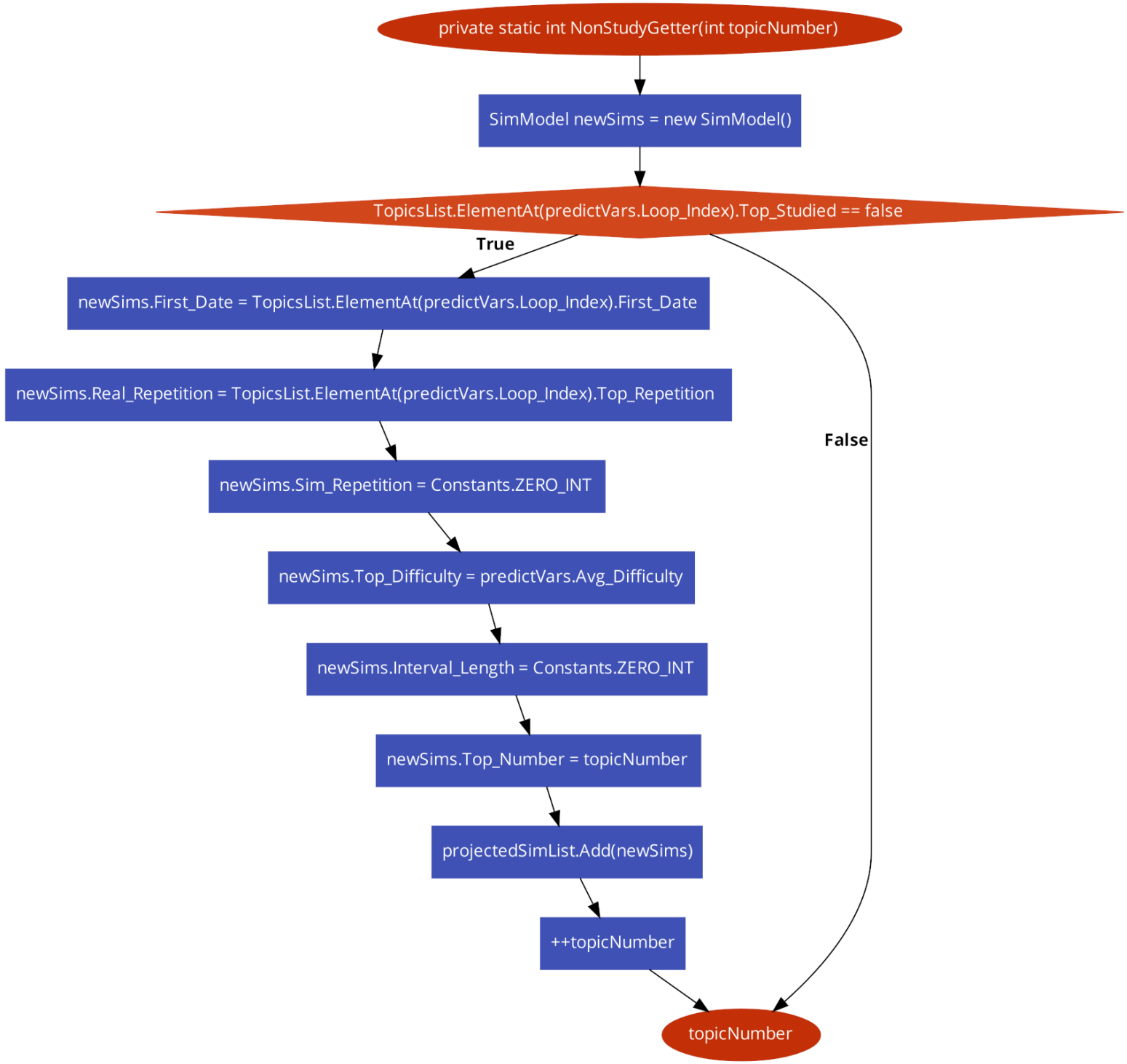

private static void xMaxCountReps()
int firstRep = Constants.ONE_INT
predictVars.First_Check == true
True
False
firstRep != toXmax.ElementAt(predictVars.Loop_Index).Sim_Repetition
True
False
xRepDateCounts.Add(Constants.ZERO_INT)
xRepTempDates.Add(toXmax.ElementAt(predictVars.Loop_Index).Repetition_Date)
predictVars.First_Check = false
predictVars.First_Check == false
True
firstRep != toXmax.ElementAt(predictVars.Loop_Index).Sim_Repetition
True
xRepTempDates.ElementAt(predictVars.XrepIndex) == toXmax.ElementAt(predictVars.Loop_Index).Repetition_Date
True
False
++xRepDateCounts
++predictVars.XrepIndex
xRepTempDates.Add(toXmax.ElementAt(predictVars.Loop_Index).Repetition_Date)
xRepDateCounts.Add(Constants.ONE_INT)

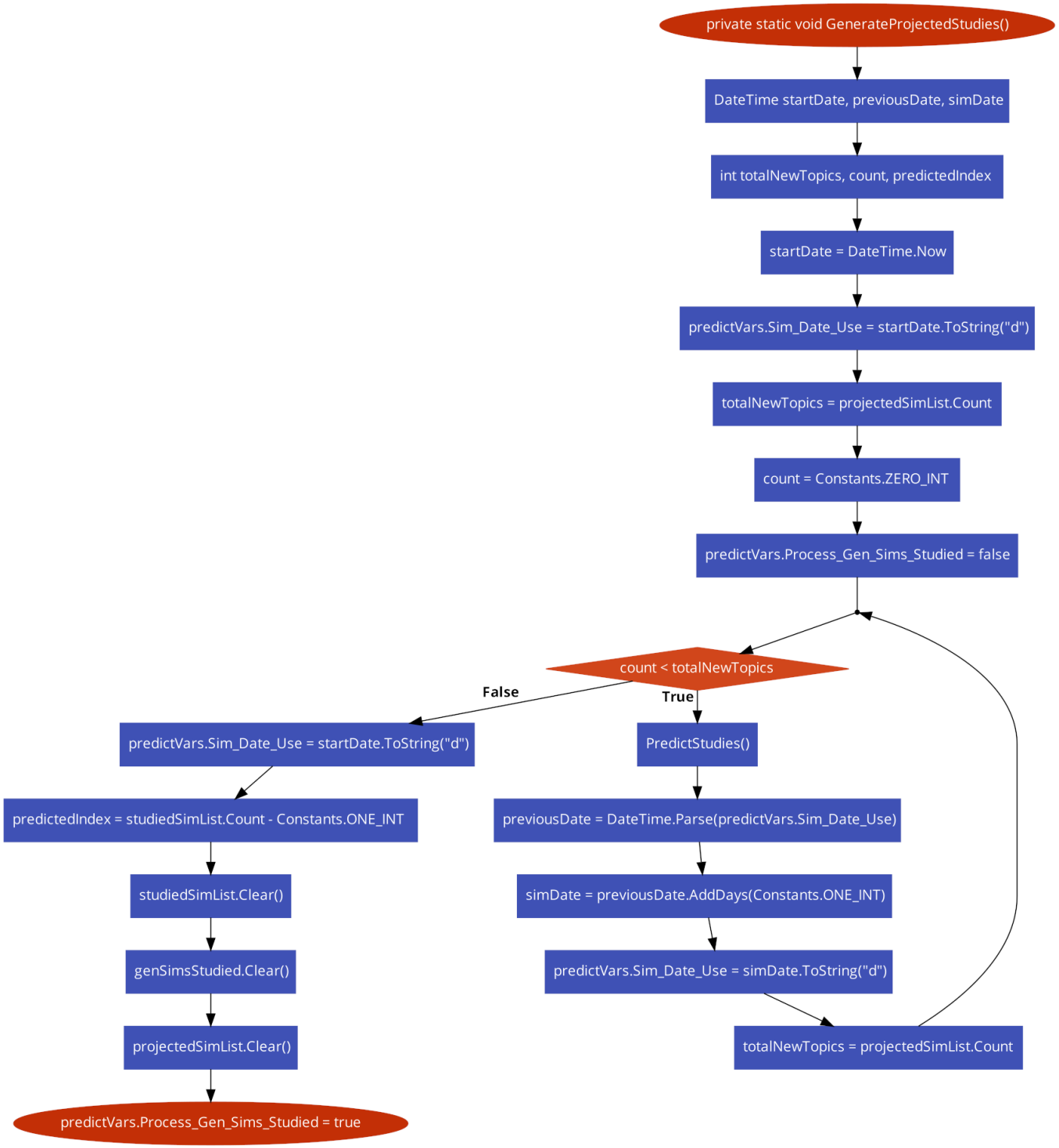

private static void XmaxRepeatSort()
predictVars.J = Constants.ZERO_INT
predictVars.I = Constants.ZERO_INT
predictVars.Date_Check = Constants.ZERO_INT
predictVars.Xsort_Count = Constants.ZERO_INT
predictVars.Xsort_Count < Constants.TWO_INT
False
True
predictVars.J = Constants.ZERO_INT
XinsertSort()
predictVars.I = Constants.ZERO_INT
++predictVars.Xsort_Count
predictVars.Date_Check = Constants.ZERO_INT
predictVars.Xsort_Count = Constants.ZERO_INT

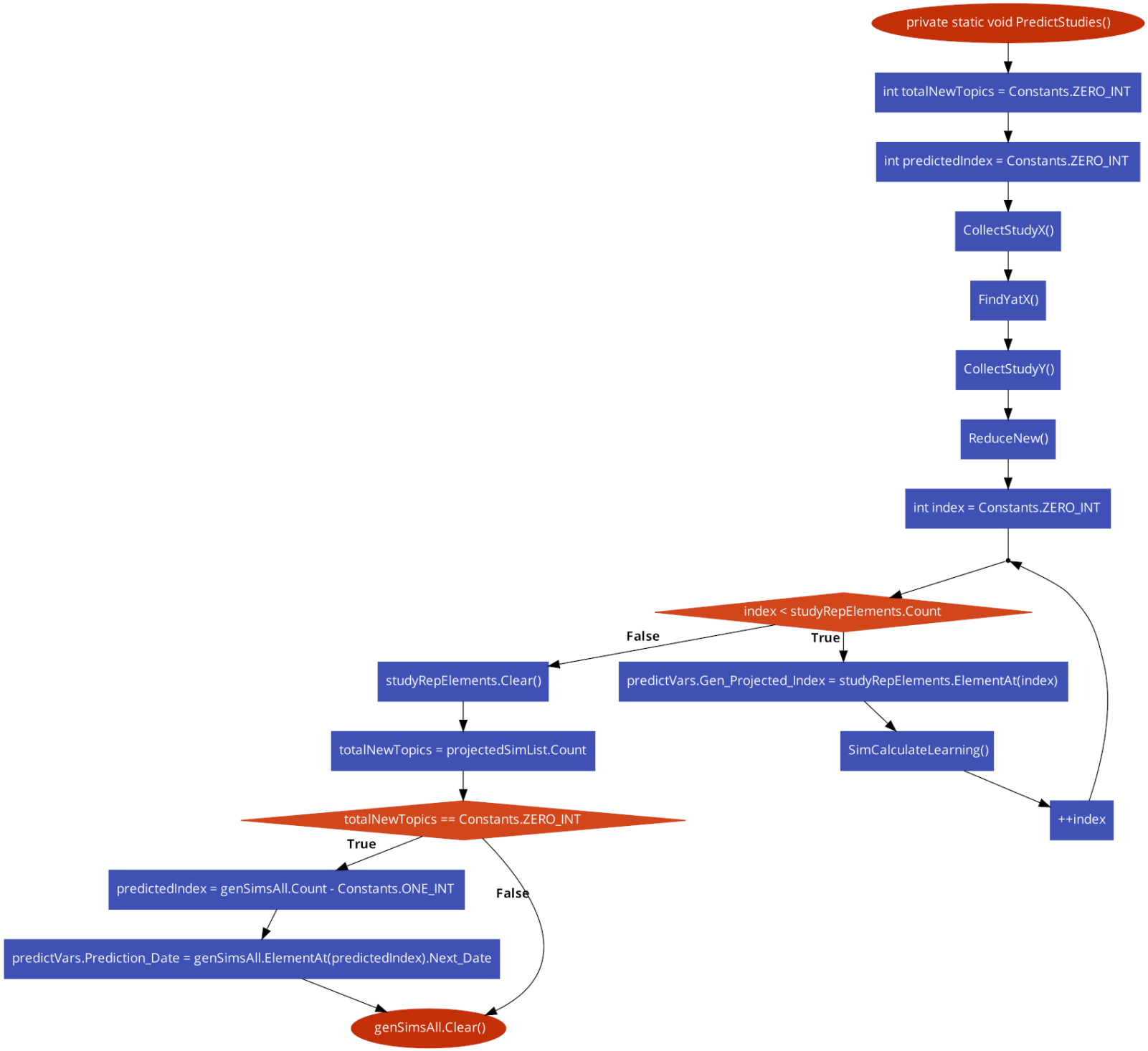

private static void XinsertSort()
SimModel listKey = new SimModel()
predictVars.J = Constants.TWO_INT
predictVars.J < xMaxSortList.Count
False
True
predictVars.Xsort_Count == Constants.ZERO_INT
True
listKey = xMaxSortList
predictVars.J
listKey = xMaxSortList
Constants.ZERO_INT
predictVars.I = predictVars.J - Constants.ONE_INT
xMaxSortList[Constants.ZERO_INT] = xMaxSortList
predictVars.Date_Check = DateTime.Compare(DateTime.Parse(xMaxSortList.ElementAt(predictVars.I).Repetition_Date), DateTime.Parse(listKey.Repetition_Date))
Constants.ONE_INT
xMaxSortList[Constants.ONE_INT] = listKey
predictVars.I > Constants.ZERO_INT && predictVars.Date_Check > Constants.ZERO_INT
True
False
xMaxSortList[predictVars.I + Constants.ONE_INT] = xMaxSortList
xMaxSortList[predictVars.I + Constants.ONE_INT] = listKey
predictVars.I
predictVars.I = predictVars.I - Constants.ONE_INT
predictVars.J++

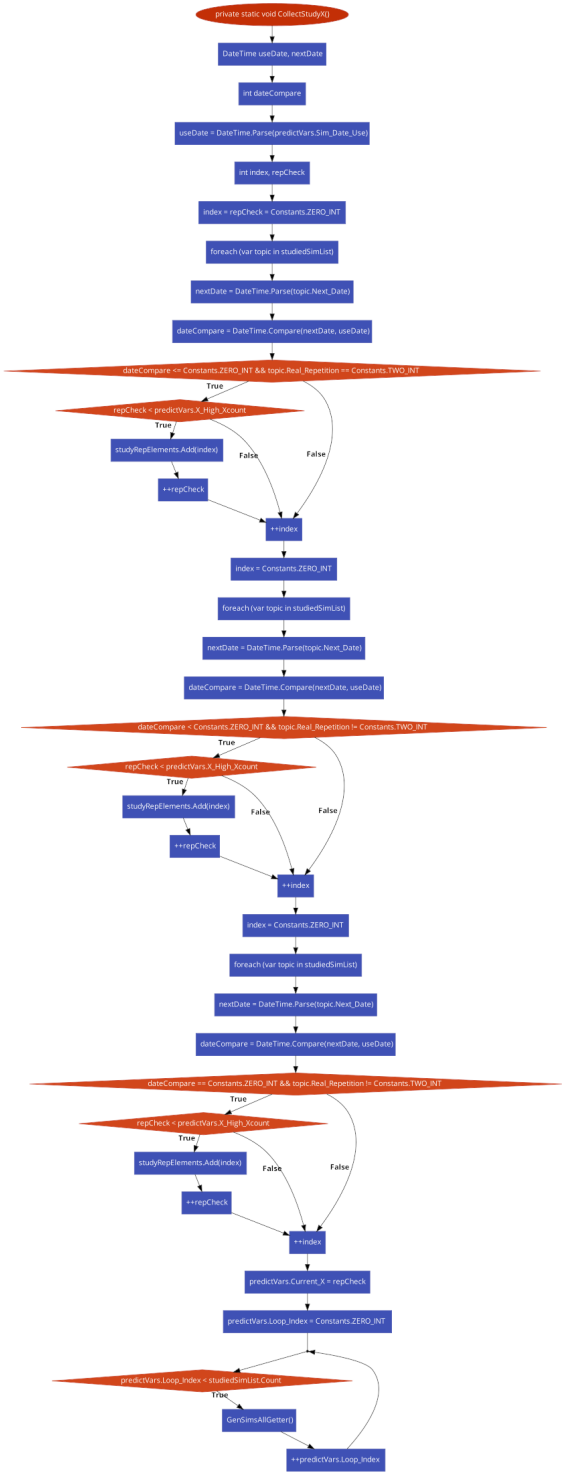

private static void XmaxFirsts()
DateTime topicDate
DateTime xMaxDate
int dateCompare
int index = Constants.ZERO_INT
index < xMaxList.Count
False
True
double check = Constants.ZERO_INT
int useable = Constants.ZERO_INT
index = Constants.ZERO_INT
check = xMaxList.ElementAt(index).Y_Count
xMaxDate = DateTime.Parse(xMaxList.ElementAt(index).High_Date)
foreach (var topic in genSimsStudied)
topicDate = DateTime.Parse(topic.Repetition_Date)
dateCompare = DateTime.Compare(topicDate, xMaxDate)
topic.Sim_Repetition == Constants.ONE_INT && dateCompare == Constants.ZERO_INT
True
False
++xMaxList.ElementAt(index).Y_Count
++index
index < xMaxList.Count
False
True
predictVars.X_High_Ycount = xMaxList.ElementAt(useable).Y_Count
predictVars.X_High_Xcount = xMaxList.ElementAt(useable).X_Count
(index + Constants.ONE_INT) < xMaxList.Count
True
check < xMaxList.ElementAt(index + Constants.ONE_INT).Y_Count
True
check = xMaxList.ElementAt(index + Constants.ONE_INT).Y_Count
useable = index + Constants.ONE_INT
False
False
++index

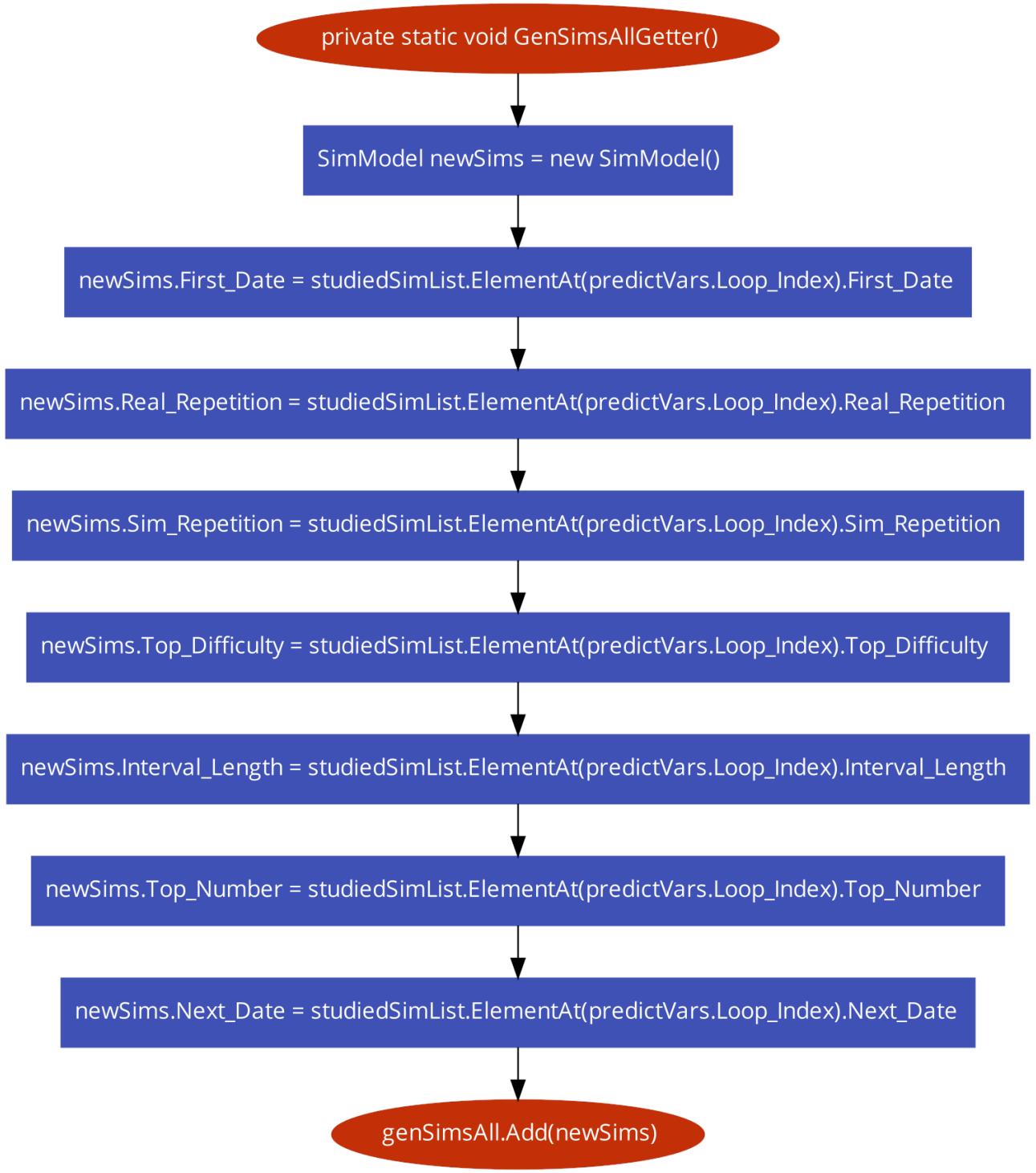

private static void CollectNonStudied()
int index = studiedSimList.Count - Constants.ONE_INT
int topicNumber = studiedSimList.ElementAt(index).Top_Number + Constants.ONE_INT
int tracker = Constants.ZERO_INT
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.Loop_Index < TopicsList.Count
True
False
tracker = NonStudyGetter(topicNumber)
predictVars.Loop_Index = Constants.ZERO_INT
topicNumber = tracker
++predictVars.Loop_Index

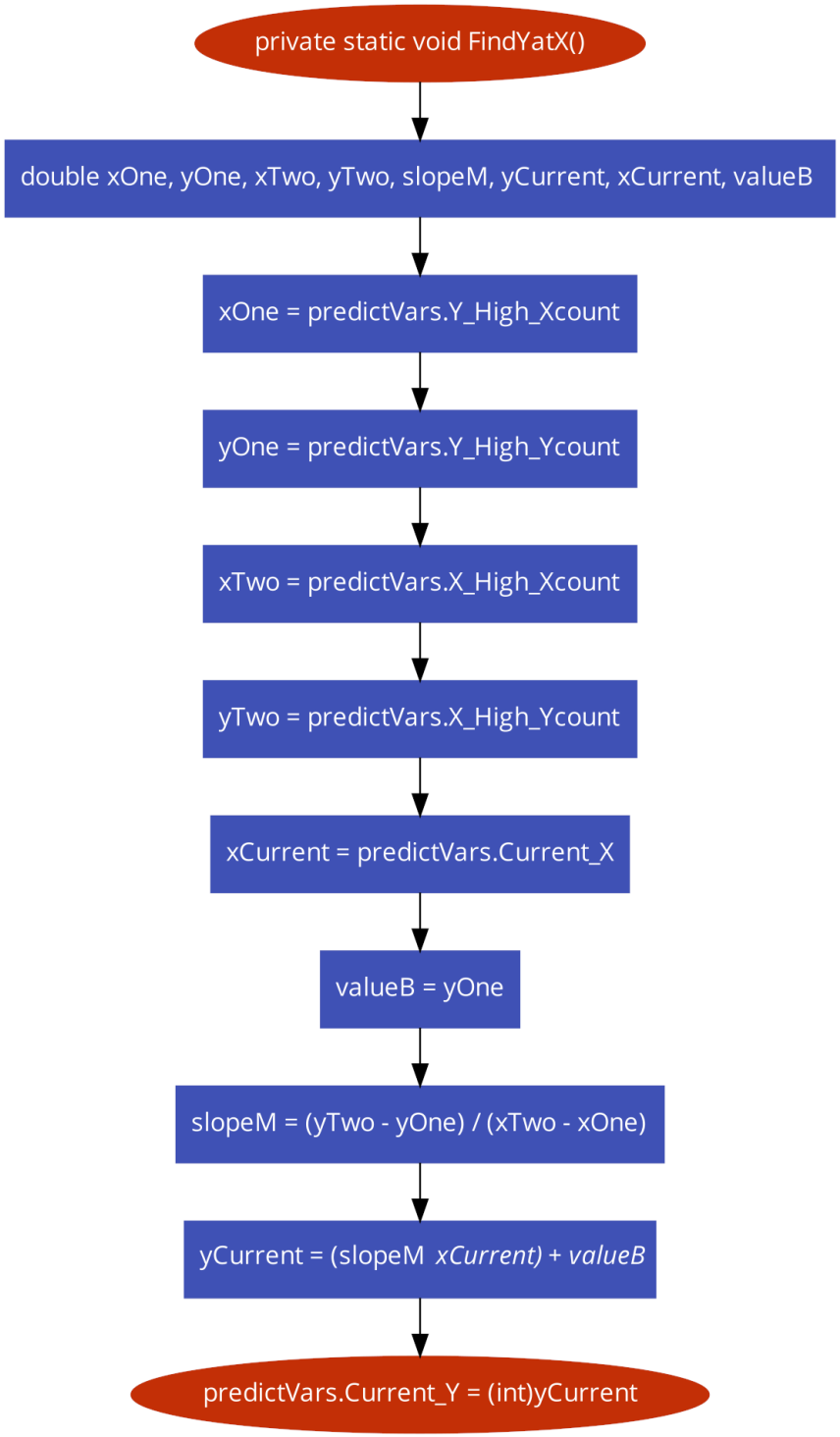

private static int NonStudyGetter(int topicNumber)
SimModel newSims = new SimModel()
TopicsList.ElementAt(predictVars.Loop_Index).Top_Studied == false
True
False
newSims.First_Date = TopicsList.ElementAt(predictVars.Loop_Index).First_Date
newSims.Real_Repetition = TopicsList.ElementAt(predictVars.Loop_Index).Top_Repetition
newSims.Sim_Repetition = Constants.ZERO_INT
newSims.Top_Difficulty = predictVars.Avg_Difficulty
newSims.Interval_Length = Constants.ZERO_INT
newSims.Top_Number = topicNumber
projectedSimList.Add(newSims)
++topicNumber
topicNumber

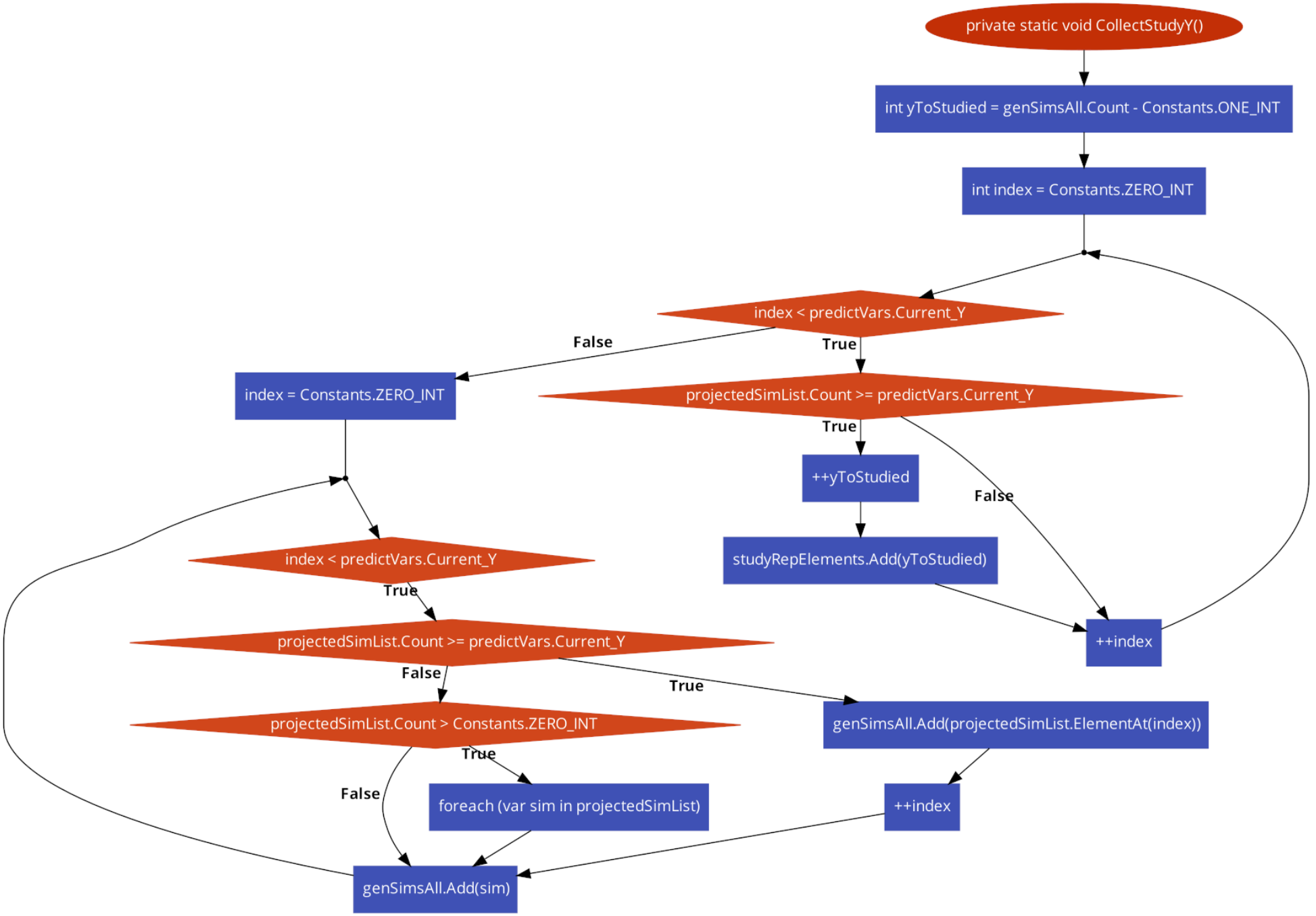

private static void GenerateProjectedStudies()
DateTime startDate, previousDate, simDate
int totalNewTopics, count, predictedIndex
startDate = DateTime.Now
predictVars.Sim_Date_Use = startDate.ToString("d")
totalNewTopics = projectedSimList.Count
count = Constants.ZERO_INT
predictVars.Process_Gen_Sims_Studied = false
count < totalNewTopics
False
True
predictVars.Sim_Date_Use = startDate.ToString("d")
predictedIndex = studiedSimList.Count - Constants.ONE_INT
studiedSimList.Clear()
genSimsStudied.Clear()
projectedSimList.Clear()
predictVars.Process_Gen_Sims_Studied = true
PredictStudies()
previousDate = DateTime.Parse(predictVars.Sim_Date_Use)
simDate = previousDate.AddDays(Constants.ONE_INT)
predictVars.Sim_Date_Use = simDate.ToString("d")
totalNewTopics = projectedSimList.Count

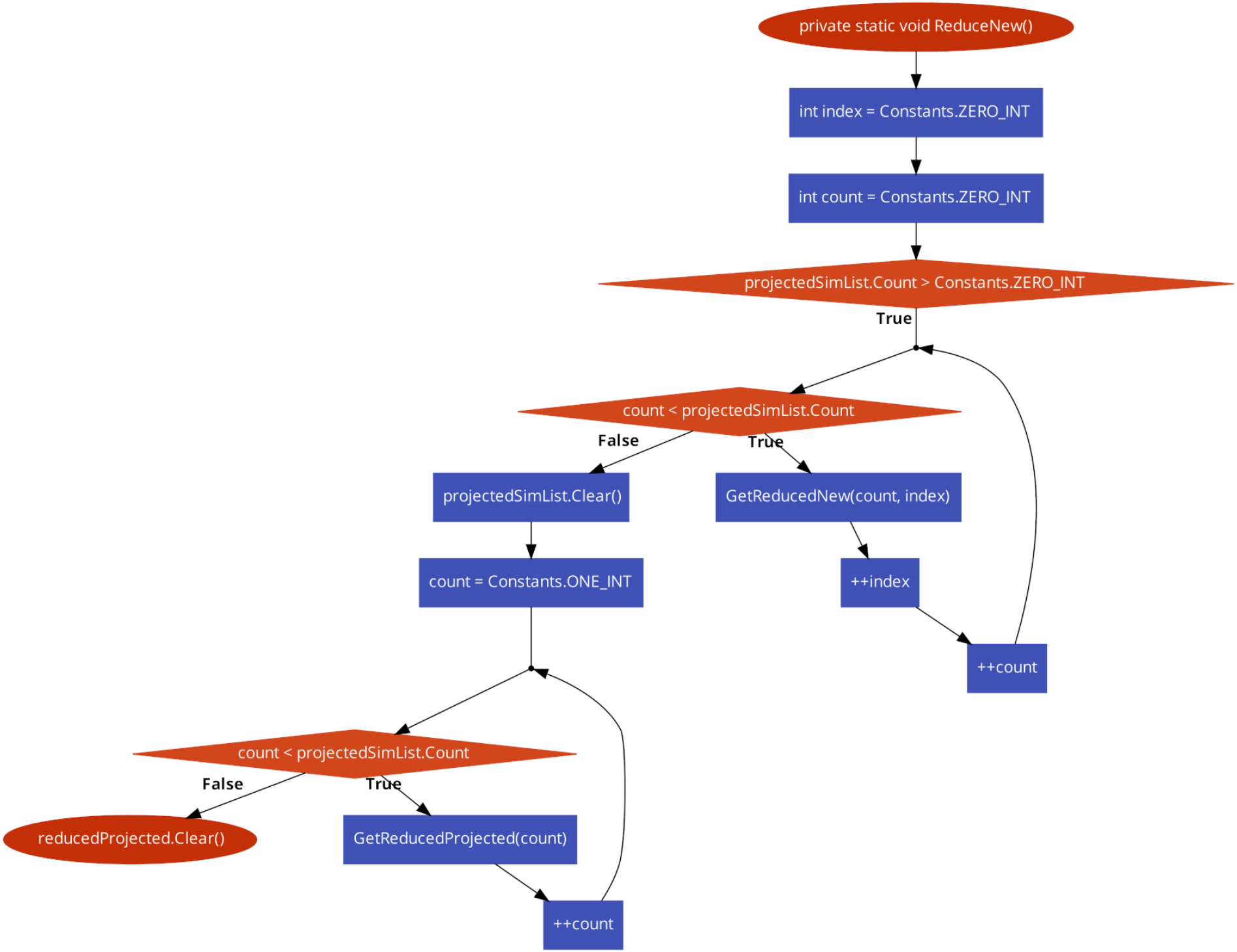

private static void PredictStudies()
int totalNewTopics = Constants.ZERO_INT
int predictedIndex = Constants.ZERO_INT
CollectStudyX()
FindYatX()
CollectStudyY()
ReduceNew()
int index = Constants.ZERO_INT
index < studyRepElements.Count
False
True
studyRepElements.Clear()
predictVars.Gen_Projected_Index = studyRepElements.ElementAt(index)
totalNewTopics = projectedSimList.Count
SimCalculateLearning()
totalNewTopics == Constants.ZERO_INT
++index
True
False
predictedIndex = genSimsAll.Count - Constants.ONE_INT
predictVars.Prediction_Date = genSimsAll.ElementAt(predictedIndex).Next_Date
genSimsAll.Clear()

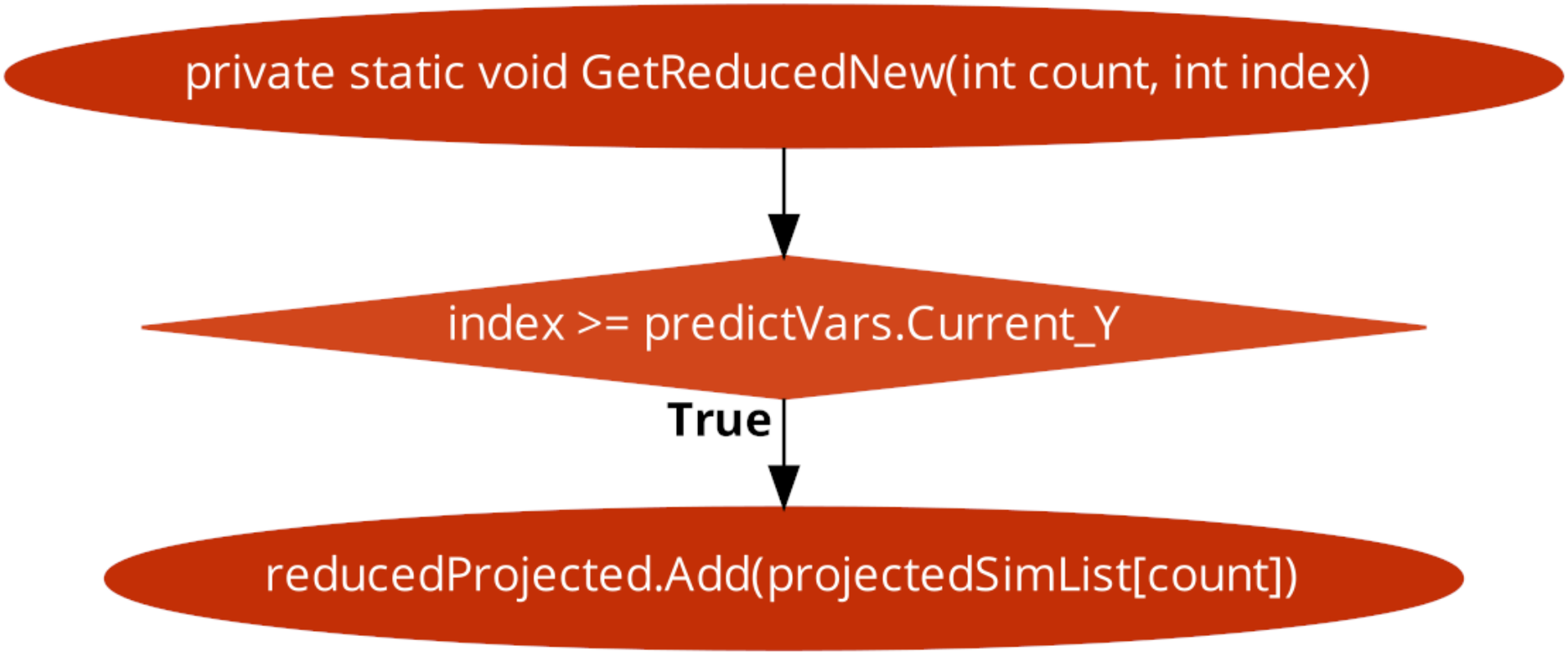

private static void CollectStudyX()
DateTime useDate, nextDate
int dateCompare
useDate = DateTime.Parse(predictVars.Sim_Date_Use)
int index, repCheck
index = repCheck = Constants.ZERO_INT
foreach (var topic in studiedSimList)
nextDate = DateTime.Parse(topic.Next_Date)
dateCompare = DateTime.Compare(nextDate, useDate)
dateCompare <= Constants.ZERO_INT && topic.Real_Repetition == Constants.TWO_INT
True
repCheck < predictVars.X_High_Xcount
True
studyRepElements.Add(index)
False
False
++repCheck
++index
index = Constants.ZERO_INT
foreach (var topic in studiedSimList)
nextDate = DateTime.Parse(topic.Next_Date)
dateCompare = DateTime.Compare(nextDate, useDate)
dateCompare < Constants.ZERO_INT && topic.Real_Repetition != Constants.TWO_INT
True
repCheck < predictVars.X_High_Xcount
True
studyRepElements.Add(index)
False
False
++repCheck
++index
index = Constants.ZERO_INT
foreach (var topic in studiedSimList)
nextDate = DateTime.Parse(topic.Next_Date)
dateCompare = DateTime.Compare(nextDate, useDate)
dateCompare == Constants.ZERO_INT && topic.Real_Repetition != Constants.TWO_INT
True
repCheck < predictVars.X_High_Xcount
True
studyRepElements.Add(index)
False
False
++repCheck
++index
predictVars.Current_X = repCheck
predictVars.Loop_Index = Constants.ZERO_INT
predictVars.Loop_Index < studiedSimList.Count
True
GenSimsAllGetter()
++predictVars.Loop_Index

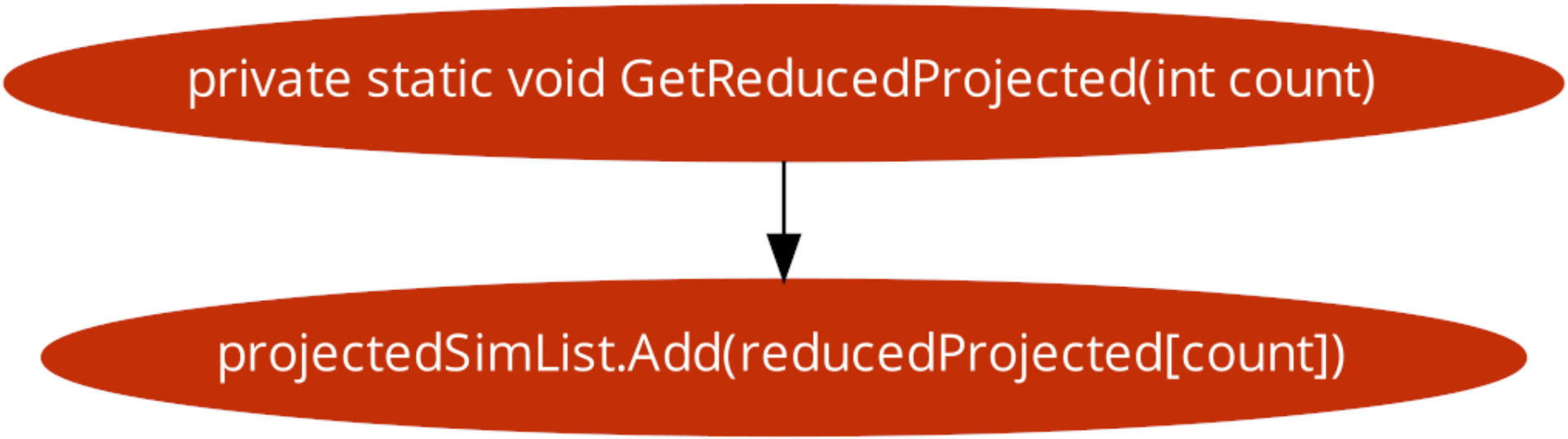

private static void GenSimsAllGetter()
SimModel newSims = new SimModel()
newSims.First_Date = studiedSimList.ElementAt(predictVars.Loop_Index).First_Date
newSims.Real_Repetition = studiedSimList.ElementAt(predictVars.Loop_Index).Real_Repetition
newSims.Sim_Repetition = studiedSimList.ElementAt(predictVars.Loop_Index).Sim_Repetition
newSims.Top_Difficulty = studiedSimList.ElementAt(predictVars.Loop_Index).Top_Difficulty
newSims.Interval_Length = studiedSimList.ElementAt(predictVars.Loop_Index).Interval_Length
newSims.Top_Number = studiedSimList.ElementAt(predictVars.Loop_Index).Top_Number
newSims.Next_Date = studiedSimList.ElementAt(predictVars.Loop_Index).Next_Date
genSimsAll.Add(newSims)

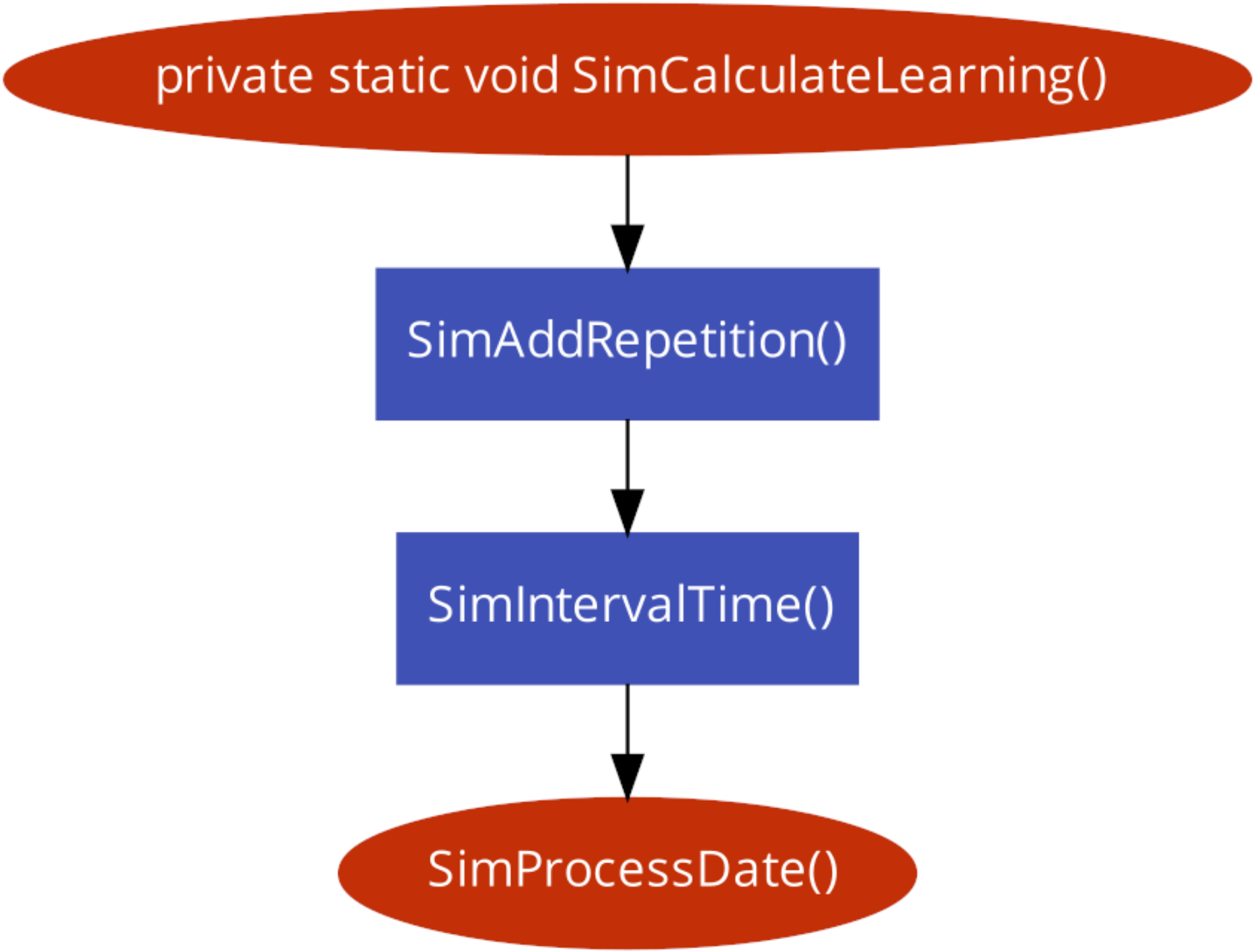

private static void FindYatX()
double xOne, yOne, xTwo, yTwo, slopeM, yCurrent, xCurrent, valueB
xOne = predictVars.Y_High_Xcount
yOne = predictVars.Y_High_Ycount
xTwo = predictVars.X_High_Xcount
yTwo = predictVars.X_High_Ycount
xCurrent = predictVars.Current_X
valueB = yOne
slopeM = (yTwo - yOne) / (xTwo - xOne)
yCurrent = (slopeM xCurrent) + valueB
predictVars.Current_Y = (int)yCurrent

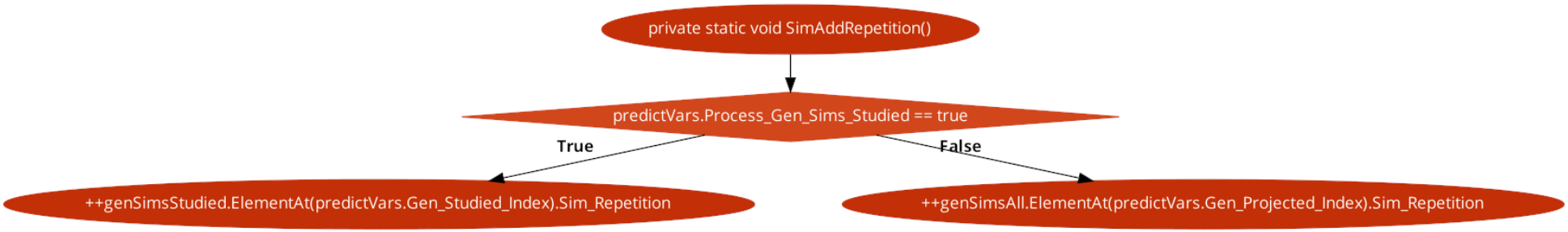

private static void CollectStudyY()
int yToStudied = genSimsAll.Count - Constants.ONE_INT
int index = Constants.ZERO_INT
index < predictVars.Current_Y
False
True
index = Constants.ZERO_INT
projectedSimList.Count >= predictVars.Current_Y
True
++yToStudied
False
studyRepElements.Add(yToStudied)
index < predictVars.Current_Y
True
++index
projectedSimList.Count >= predictVars.Current_Y
False
True
projectedSimList.Count > Constants.ZERO_INT
genSimsAll.Add(projectedSimList.ElementAt(index))
True
False
foreach (var sim in projectedSimList)
++index
genSimsAll.Add(sim)

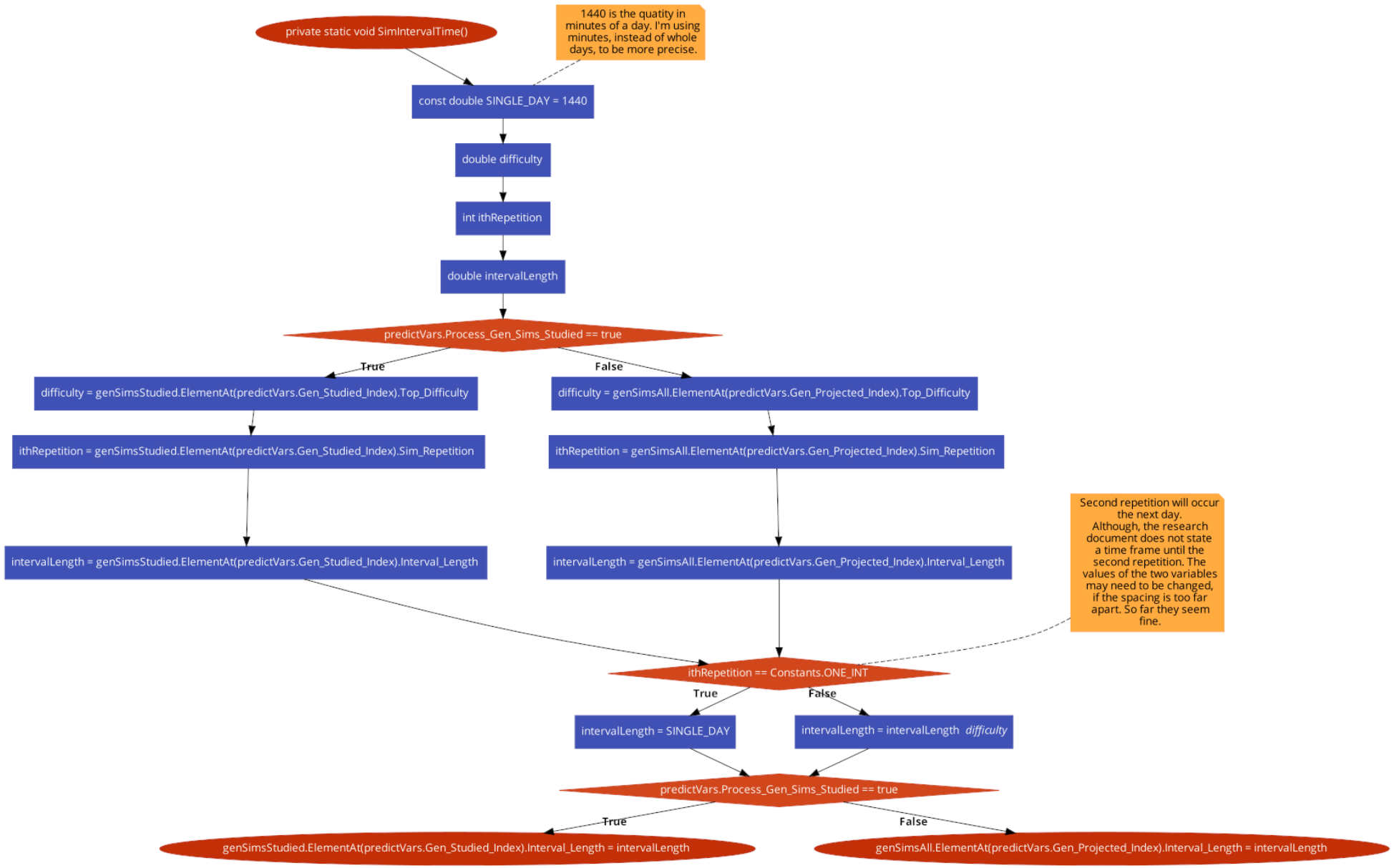

private static void ReduceNew()
int index = Constants.ZERO_INT
int count = Constants.ZERO_INT
projectedSimList.Count > Constants.ZERO_INT
True
count < projectedSimList.Count
False
True
projectedSimList.Clear()
GetReducedNew(count, index)
count = Constants.ONE_INT
++index
++count
count < projectedSimList.Count
False
True
reducedProjected.Clear()
GetReducedProjected(count)
++count

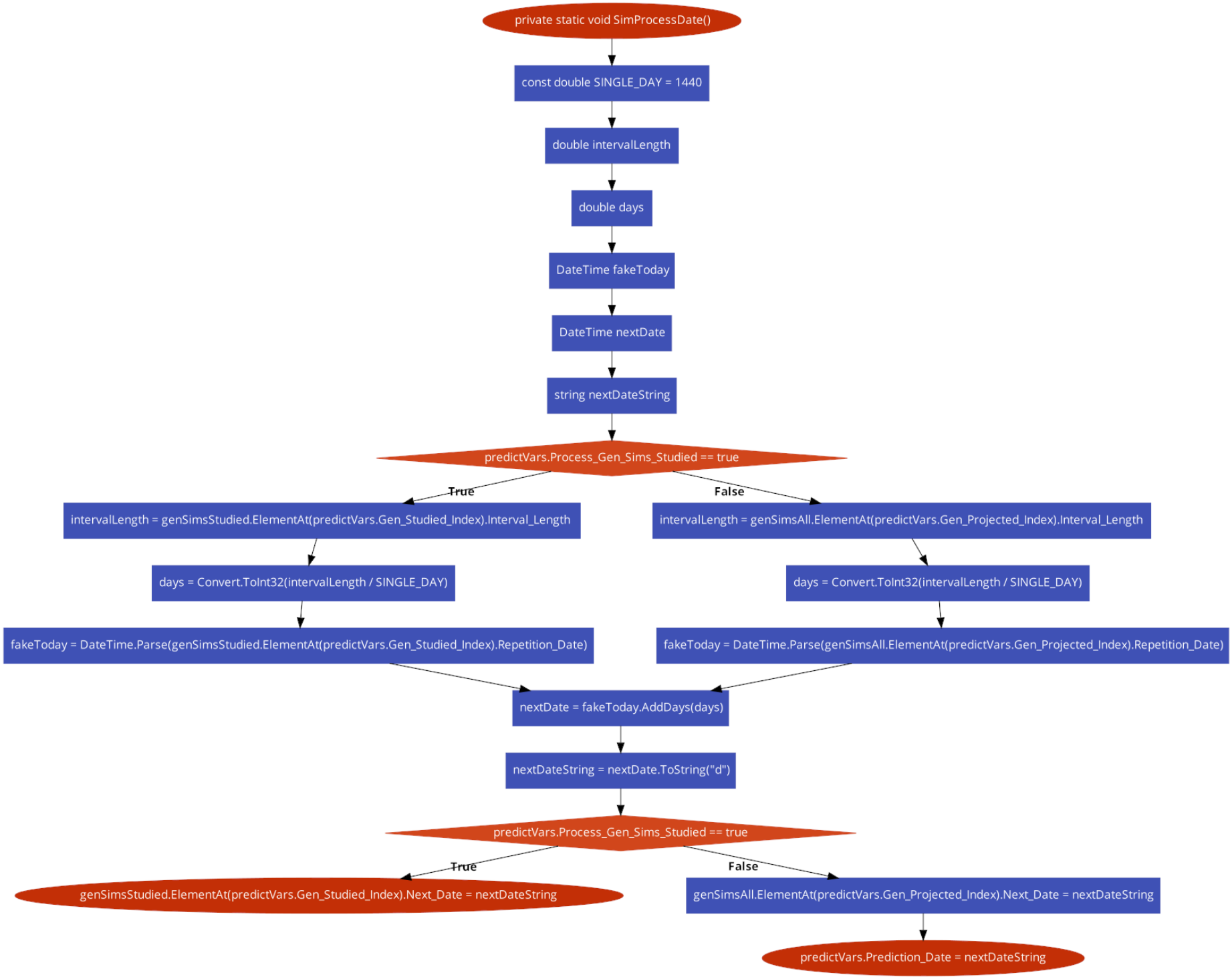

private static void GetReducedNew(int count, int index)
index >= predictVars.Current_Y
True
reducedProjected.Add(projectedSimList[count])


private static void GetReducedProjected(int count)
projectedSimList.Add(reducedProjected[count])


private static void SimCalculateLearning()
SimAddRepetition()
SimIntervalTime()
SimProcessDate()


private static void SimAddRepetition()
predictVars.Process_Gen_Sims_Studied == true
True
False
++genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Sim_Repetition
++genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Sim_Repetition


private static void SimIntervalTime()
1440 is the quatity in minutes of a day. I'm using minutes, instead of whole days, to be more precise.
const double SINGLE_DAY = 1440
double difficulty
int ithRepetition
double intervalLength
predictVars.Process_Gen_Sims_Studied == true
True
False
difficulty = genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Top_Difficulty
difficulty = genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Top_Difficulty
ithRepetition = genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Sim_Repetition
ithRepetition = genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Sim_Repetition
intervalLength = genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Interval_Length
intervalLength = genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Interval_Length
Second repetition will occur the next day. Although, the research document does not state a time frame until the second repetition. The values of the two variables may need to be changed, if the spacing is too far apart. So far they seem fine.
ithRepetition == Constants.ONE_INT
True
False
intervalLength = SINGLE_DAY
intervalLength = intervalLength difficulty
predictVars.Process_Gen_Sims_Studied == true
True
False
genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Interval_Length = intervalLength
genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Interval_Length = intervalLength


private static void SimProcessDate()
const double SINGLE_DAY = 1440
double intervalLength
double days
DateTime fakeToday
DateTime nextDate
string nextDateString
predictVars.Process_Gen_Sims_Studied == true
True
False
intervalLength = genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Interval_Length
intervalLength = genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Interval_Length
days = Convert.ToInt32(intervalLength / SINGLE_DAY)
days = Convert.ToInt32(intervalLength / SINGLE_DAY)
fakeToday = DateTime.Parse(genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Repetition_Date)
fakeToday = DateTime.Parse(genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Repetition_Date)
nextDate = fakeToday.AddDays(days)
nextDateString = nextDate.ToString("d")
predictVars.Process_Gen_Sims_Studied == true
True
False
genSimsStudied.ElementAt(predictVars.Gen_Studied_Index).Next_Date = nextDateString
genSimsAll.ElementAt(predictVars.Gen_Projected_Index).Next_Date = nextDateString
predictVars.Prediction_Date = nextDateString