A COMPUTER PROGRAM FOR TRACKING CANCER DEVELOPMENT AND MOVEMENT

NEW MEXICO ADVENTURES IN
SUPERCOMPUTING CHALLENGE
FINAL REPORT
APRIL 6, 2004

TEAM 004
ALBUQUERQUE ACADEMY

Project Members:
Punit Shah
Karalyn Baca

Teacher:
Jim Mims

Project Mentor:
Gene Wong, M.D.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

Nearly seventy-five percent of all skin cancer deaths are caused by melanoma, making it the most serious type of skin cancer.[1] While melanoma has the more predictable growth pattern of basic types of skin cancer, there is still a factor of randomness in its growth. For these reasons, this project attempts to simulate the growth of malignant melanoma cells taking into account as many factors as possible to simulate real-world situations. Factors include competition for nutrients between malignant cells and competition for the infection of healthy cells. This project attempts to come closer to the ideal model of malignant melanoma cancer simulation and gain a greater understanding of this deadly cancer.

In order to model melanoma cancer growth in a meaningful manner, we needed a highly graphical programming language and therefore we chose Microsoft Visual Basic 6.0. It allowed for the relatively easy production of the spectacular graphics needed and allowed for us to concentrate more on the algorithms and virtual model.

The model used a simple, yet very effective and relatively efficient "sweep" method. The process has the program go through the organized representation of the skin cross-section model and develop the cancer according to the parameters and facts that researchers have discovered in labs and case studies. The program factors in probability as well as the pattern of melanoma growth, such as horizontally across the skin and physical conditions of the cancer cells. The program then repeats this "sweep" process multiple times as it continues to develop the cancer through the model.

Our results coincided with test data, which shows that cancer initially follows exponential growth pattern, and then levels out as time progresses. This pattern is called a "sigmoidal" curve. Because our program tends to follow this curve, it appears that the produced results are quite accurate, a major achievement of the project

[1] American Academy of Dermatology, "Melanoma Fact Sheet." <www.aad.org>

INTRODUCTION

In choosing a project, we wanted to select something that could have a positive effect on others, such as a program that contributed to research in a certain life-enriching field, such as medicine. In the end, we arrived at modeling the spread and growth of melanoma skin cancer. Hopefully this program will assist in understanding the movement of melanoma cancer so it can be treated more effectively and make even a minor contribution to our modern understanding of this deadly cancer. By creating a graphical and textual representation of the cancer's movement, it would be possible to anticipate the cancer's movement and prevent it from spreading. This problem would definitely need the resources of a supercomputer as the model developed in complexity, and would therefore be very appropriate for the AiS Challenge.

There are three basic types of skin cancer, squamous cell carcinoma, melanoma, and basal cell carcinoma or epithelioma. We chose melanoma because of its relatively predictable movement allowing for a model of greater precision; its movement is the most predictable of the three types of skin cancer, even though none are totally predictable.

Skin cancer is formed when healthy melanocytes, the pigment producing cells in skin, are exposed to excess amounts of UV rays. The DNA in the nucleus of the cell is then damaged, causing for the cancerous "mutant" cell to develop. Melanocytes reside in the area called the germinal layer, between the epidermis and dermis (See Appendix A, Figure 1) and have attempted to restrict growth to this general region. These abnormal cells begin to behave strangely and are now considered melanoma.

These damaged melanoma cells produce excess melanin, the pigment that colors one's skin, causing the skin to become darker in the infected area (See Appendix A, Figure 2) and the

cells also grow abnormally large, breaking through the epidermis. Melanoma develops in one area and spreads horizontally due to the pressure from the other skin layers, giving it an elliptical shape. As these malignant melanocytes grow, they consume healthy red blood cells which they use for growth as well as destroying local cells like bone tissues and local capillaries. The body attempts to replenish the blood supply at these locations where the malignant melanocytes have consumed them. Excess bleeding, therefore, is a common symptom in patients with melanoma.

The malignant cells gather together and form colonies, or moles. When these colonies reach approximately 5 mm, cells tend to break off and form satellite nodules, or other moles close yet separated from the original colony. As development continues and the cancerous cells start to leak into the arteries, satellite nodules can eventually travel through the lymphatic channels (See Appendix A, Figure 3), the body's filtering system. The "filters" catch the melanoma cells and while the cancerous cells get trapped they continue to grow. As a result, patients with melanoma often complain of pain in their armpits, groin, and neck, the locations of major lymph glands, the actual filters. The melanoma cells caught in these filters continue to grow and exert pressure on their surroundings, causing discomfort. Once in the final stages of growth, malignant melanocytes are able to travel through the blood stream, which reaches all parts of the body. This causes blockages where they are caught, especially in such places as the heart, lungs, and brain, where they can be fatal.

Despite the frightful nature of this cancer, treatment is possible when detected early before the cancerous cells spread through the blood stream and body. Models such as the one developed in this project would be very helpful in treating the cancer.

DESCRIPTION AND METHOD

This program strives to model melanoma cancer's growth through the body taking into account the many factors that influence its growth. In our initially simplified model, the cancer originates at certain coordinates and from there follows an elliptical growth pattern. As time progresses, the cancerous cell count compared to the time should follow a sigmoidal curve, a curve that initially resembles a representation of exponential growth but then in time levels out **(**Appendix A, Figure 4). Initially, there will be exponential growth as there is virtually no competition for resources and space, which later becomes a problem. At first there is nothing stopping these malignant cells' growth. As time progresses, though, resources become a problem for development and therefore growth is slowed and levels off.

Gene Wong, M.D., our project mentor recommended that we attempt to make a program that initially follows reality and then check to make sure that various other cases worked in this scenario, editing the program so they do. This is the method used in developing professional computer models of cancer growth and has been somewhat successful. For that reason, we decided to try that method in developing our program.

Before starting on the application, we had to select a programming language. After looking at the pros and the cons of each language, Microsoft Visual Basic 6.0 appeared to fit the needs of the project quite well. As the name suggests, it is very graphically oriented, something that would be important for this program. Therefore, we used it to easily create excellent graphics while allowing us to concentrate much more on the algorithms, the part of the project that is most important. For a simple, concise view of our algorithm and general approach, see the flowchart in Appendix A, Figure 5.

To model the cancer cells we decided to have each pixel on the screen represent a melanocyte cell. The number of pixels are user defined, depending on how fine of a resolution the user wants. The healthy cells are represented by a gray dot, while the cancerous cells are red dots. Each cell is an object of our GraphicCell, which includes various properties that are changed by the different methods or sub programs. As each cell is represented by a pixel on the screen, we used a 2-D array, or basically a table containing these different human cells, named CancerModel to hold each of these objects. The name of each of the elements in the array corresponds to the x and y value of the pixel or cell on the screen, making it is somewhat like a large graph. We used a 2-D array instead of a 3-D array because we believe it provides a reasonable representation of the situation. Additionally, the third dimension would use a large amount of memory with resulting increase in run time. We accommodated various variables on each cell with the use of object-oriented programming. The variables for each cell were stored in objects of the class used.

Object Oriented Programming, or OOP, is a method of programming that makes “objects” of a “class” and then manipulates these objects for programmer’s purpose. The reason this is so useful for this project is that we are able to make the components of each of the cells uniform and then manipulate them, for instance making some or all of them cancerous. We are also able to check the amount of available nutrients, a factor in growth, there are for a certain cell by calling a function that calculates and returns the amount of nutrients. Also, it is easily able to be expanded upon when giving the program greater abilities by adding more properties to the class without influencing the rest of the program. These are only a few examples of how OOP gives us a much greater opportunity to utilize all the cells of our 2-D array of objects, each cell with a member of our class GraphicCell.

The class that we created is called GraphicCell. It contains the different properties for each of the objects of the class located in the 2-D array, including IsCancerous (the property stating whether that cell of the array had been infected), HasBeenSwept (used for tabulation of cancerous cells which is described later), nutrients (amount of available nutrients to that specific cell), and other similar properties. These different properties are manipulated by the driving classes to make the model work. Also, the way in which the class is used by the program and designed allows for it to be easily expanded. This becomes very important as we continue to develop the program because when starting initially, we wanted few factors to be incorporated so the problem would be simplified, though as we progress through the project and wanted more factors be influence growth, we were able to just add to the class.

The major method in our program is the sub procedure InitiateAction, which calls many of the other function procedures (also called “procedures”) and sub procedures (also called “subs”). In Visual Basic (VB), a function procedure returns a value to the calling procedure and subs do not. Among these subs are SweepTabulate, SweepGrow, and SweepReset. Each of these subs “sweeps” through the array CancerModel and changes certain properties of the object in the array. SweepTabulate checks each object in the array to see if the property IsCancerous, a boolean, is set to true. If so, the sub will then set another of the object’s properties, HasBeenSwept to true. In the next sweep, SweepGrow, the sub determines which objects of the array have the HasBeenSwept property equal to true; if this is the case, the program calls the sub InfectNeighbors, passing it the name of the cancerous pixel or cell. The sub InfectNeighbors uses these passed values, which correspond to the cancerous cell’s location on the x-y plane, to calculate its neighbors. After each neighbor is calculated, the sub then calls the function ShouldIInfect.

ShouldIInfect returns a true or false value to determine whether the cancerous cell should infect its nearby healthy cells. This procedure really emphasizes that growth is a function of the cancer cell's location, initial size, nutrients, etc. In this function, ShouldIInfect, we can add many types of cases, random number generators, and equations to determine whether the cancerous cell infects healthy cells, though at first we kept these relatively simple for error checking purposes and keeping the problem relatively simple, a good practice to start with. Currently, some of the factors that are taken into account include skin layer pressure, a major one, general probability of growth, and others factors. The ShouldIInfect sub works using a counter of the odds of infection, which is influenced differently by specific factors, these factors will make it more unlikely or likely for the cancer to develop in a certain spot. For example, when the cancer starts to get quite high and there is a lot of skin layer pressure, the chance for that cell to develop up or down from that point is about 1/17. Once the odds have been added together into a counter variable, a random number is then generated from 0 to whatever the counter is at. Only if the random number is zero will the program return a "true" value for the InfectNeighbors sub to infect the specified neighbor. By having these sorts of influences, the cancer mass tends to take an oval shape (Appendix B, Figure 1), which is how melanoma cancer usually develops.

In the previous paragraph, we used the word infect to describe cancer growth, though there is a important yet basically true assumption that we are making is saying this. In real life, cancer cells do not infect the cells close by it but rather the mole just expands, though as our model is a 2-D simplification, we can accurately say that the expanding cancerous cells "take over" cells of the model which is then represented by the cancerous cells. In other words, the non-cancerous cells are still there, except pushed to the side. This simplification does not seem to

degrade the results and still present non-cancerous cells which were pushed out of their positions are factored in by the pressure that it exerts onto the cancerous mole.

The third and final sub is SweepReset, which sets all objects' HasBeenSwept property to false. This ensures that all cancerous cells will have a chance to grow during the next run through the sweeps. In the previous sweep, SweepGrow, the "If" statement tested for the value of the bolean HasBeenSwept. Rather than test for IsCancerous, we tested for HasBeenSwept to make certain that cancer cells that had been newly infected, with the InfectNeighbors function were not grown again in the same sub, SweepGrow.

In the earlier stages of our program, for testing purposes, we decided to limit the number of sweeps that were executed so that we could quickly run through our code using little computing power. The "sweeps" are the "development cycles" that appear on the Input Screen of our project, where one development cycle is one run through of each of the three sweeps (SweepTabulate, SweepGrow, and SweepReset). The number of sweeps is determined by user input so that if the user wishes to see how the cancerous point infects the healthy cells over a longer period of time, they can simply input a larger number for the number of sweeps. Also, because the starting x and y values for the first cancerous point is user determined, the user can observe how the cancer infects healthy cells when nearer the epidermis rather than deep in the dermis layer.

The process in which we plotted the model onto the form and the graph of how many cancer cells vs. time was a simple process, yet very effective and efficient, very important qualities in a problem like this. To plot the skin cross-section while showing the cancer growth, the program takes the GUI refresh rate input and updates the GUI every specified number of development cycles. Using the method of refreshing every few development cycles allows us to

avoid wasting system resources in redrawing the graphic every time when only a little progress has made between the cycles. To check if the graphic should be updated, it takes the number of development cycles that have been performed and performs the mod function on it with the GUI refresh rate. If the answer is zero, then the program updates the skin model by sweeping through the array and checking the IsCancerous property. If it is true, then a red dot is made on the form. If it is not cancerous, then a grey dot is made. A black dot is made for the location of the initial cancer. Then, the layers of the skin are drawn on top. This overlay and color-coded diagram easily shows even an untrained user immediately what is generally happening.

In addition to creating the skin cross-section view, a graph of the development cycle (x) compared to the number of cancerous cells (y) is made. The graphing sub in the program is given the current coordinates to graph. It draws a line from the previous point to the new point and then sets the new point as the old point so next time the sub is called, it can draw the line from the previous graphed point. The graph is very accurate and easy to read using this method (Appendix B, Figures 2 and 3)

RESULTS

The results that are outputted in run-through of the program are shown in two ways. The primary output is the cross section of the skin (Appendix B, Figure 6). This output has a black dot for where the initial cancer was placed. It creates red squares that together form a larger red shape that represents the cancer mole. There are also the grey dots that represent the other, non-cancerous cells. On top of the drawing is the axis overlay and the skin layers division lines.

The second output is the graph (Appendix B, figure 7). It shows the x-axis represents the development cycle number while the y-axis represents the cell count at a certain development cycle. It has the labeled axes and the maximum value for each of the axes. Also, a brief description of what the graph shows is provided.

Initially, when running this code, the output was a large red square (See Appendix B, Figure 2), with its center at the initial point of infection, as determined by the user input. No factors of growth were incorporated into this preliminary output. As we continued developing the program, perfecting the algorithm, and adding factors that affected growth, the square changed to a sphere and then into the more accurate ellipse.

While visiting with our mentor Dr.Wong, he informed us that usually the number of cancer cells can be represented by a sigmoidal (See Appendix A, Figure 4) curve, which starts out like a graph of exponential growth and then levels off, as stated in earlier sections. Initially the growth was completely exponential, though the growth became closer to this ideal curve again as development proceeded (Appendix B, Figure 3).

The ideal growth that we have is initially a part of an exponential curve (See Appendix B, Figure 4). This is what we expected to find, knowing that each cancer growth sends out satellite

nodules, which in turn each send out more satellite nodules. Such growth is the very definition of exponential and accounts for the rapid spread of melanoma cancer. As growth was reduced in later development cycles we see that good leveling out of the curve (Appendix B, Figure 3).

ANALYSIS

The graphical output of our program indicated that cancer initially grows in a circular fashion and then as it grows larger, begins to take the shape of an ellipse. Our graph indicated that initially cancer growth is exponential but then slows down, causing the graph of number of cells versus time to initially appear exponential but as the time that the simulation runs increases we see that the graph is actually sigmoidal. We have benchmarked all of these aspects of cancer with Dr. Wong. This therefore shows that the program somewhat accurately measures the growth of cancer, a very important point to be extracted from our results.

In addition, the graphical model of the skin shows that the mole expands in the elliptical fashion that we hoped for. Cancer can expand in either horizontally or vertically, though probability shows that cancer grows horizontally. The same probability is applies in this program and the same results are therefore generated that again verifies the program's accuracy.

These positive results indicate that our program is on its way to becoming useful to anyone wishing to know more about the growth and behavior of melanoma cancer. Although we have only a basic version of a cancer model, the outputs we observed from our program show that our code is sound and need only be improved upon rather than changed. Our program would be sufficient to give someone a basic understanding of the movement of cancer throughout the body and its behavior in regards to infecting other healthy cells.

Overall, it seems that the results not only compared with scientist' theoretical knowledge, but it also conformed to trends in clinical and laboratory research. These results are often hard to achieve as there are many different variables acting upon melanoma cancer development that do not exist in the theories.

CONCLUSIONS

We are able to draw many conclusions from the results we achieved, especially because the results are mostly valid. Some of the conclusions that we made in doing this projects was that cancerous cell growth did indeed follow a sigmoidal curve. The initially exponential restriction-free growth eventually ate up the system's resources which made growth rate to level and become somewhat linear. This conclusion is very important as with this knowledge, we are able to determine how to treat someone with cancer because we will know approximately how many cancerous cells there are and how to deal with it. The amount of cancerous cells and knowing which stage of development the cancer is in will also tell us how severe the cancer is or will be, giving us further insight into what little we know of this deadly disease.

We also can conclude what shape melanoma cancer moles grow in. Through the research done in this project, we confirmed our beliefs that melanoma cancer's moles grew in an elliptical shape. This again would give us insight into how to treat the cancer. Dr. Wong informed us that if the cancer is not too far into development, the cancer mole can often be cut off. Using this program, we would be able to determine the stage of development and the current shape of the cancerous mole. This would assist the surgeon in knowing where to cut without making an unneeded cut and possibly making the mole burst, which would make the cancer worse. The flexibility of the program due to the many user inputs also allows for this program to be used in purposes such as this one when the output becomes totally perfected.

ACHIEVEMENTS OF THE PROJECT

The greatest achievement of our project in our topic of research was the sigmoidal curve and accurate model that we were able to create. Though this is a simplified graph of the growth of cancerous cells, it was suitable to use for our fairly basic program. It was important to realize that we were trying to model our cancer cells in a natural environment, so there were limited amounts of space and nutrients to foster the growth of our cancer cells. We had to approximate and simplify before we could try to even come close to mimicking the natural environment of the body in our computer program. Nonetheless, the program functioned correctly which was a big feet to ensure.

When we look at the biggest achievement of the project in terms of computer science, it would probably the method in which the program was created so it could easily incorporate more features and the algorithm which the application utilized. When creating the program, the code was highly modularized. This allowed for easy reading of the code and also for easy expansion. When someone decides that they would like to expand on our model, all they have to do is edit the probability procedure and change the probability counter or add procedures and variables to the class that represent different factors. In a model like this where the number of involved factors is based on the time the programmer have, an approach such as the one we took is very important.

The second achievement of the program in computer science is the algorithm used. The simple sweep method is easy to understand and is still efficient in producing the needed results. This algorithm that we used is very important and effective.

RECOMMENDATIONS

Our program could have an innumerable amount of development variables in the ShouldIInfect function, each which would influence cancer cell growth in a different way. The ShouldIInfect function should definitely be expanded if we wished to make our program further more accurate and complex. Also, our current modeling screen is relatively small, so we could certainly enlarge the area that the cancer is allowed to grow in. In doing this however, we would be using up more system resources and this could possibly cause a slower computer to crash. So, if we did decide to enlarge the scope of the cancer modeling area, we should also insert code into our program that would write certain important statistics to file. In these statistics we would include, the number of cancerous cells versus the number of healthy cells, how any sweeps had been executed, and possibly the location of the most recently infected cells, to gain an idea of where the cancer was in relation to the rest of the graph.

Our results are an accurate representation of cancer movement but a very simplified version. So from here we would expand on our program rather than change the procedures because they work adequately. When we believed the number of variable factored in to the infect process were sufficient, it would be possible to make our program even more complex and closer to reality by modeling the movement of cancer through the lymphatic channels. Our current program only shows local growth but in actuality cancer cells travel through the lymphatic channels and infect other parts of the body. This would of course be something extra to add to the program, only in its final stages of development.

Yet another thing that we could work on if we had time would be a more advanced output system. It could b e a 3-D representation in which the user would be able to travel through a gas-

looking cloud that would be the skin. In it would be another color that represented the cancer growth. This would much more accurately show the skin and give an even better perception of what was really happening. Of course this would also be a complex and therefore system resource-intensive part of our program. Therefore, if we add many of these sorts of factors, soon we will have to test our program on a much more powerful computer that the home PCs that we have been testing it on.

ACKNOWLEDGEMENTS AND CITATIONS

We would like to thank Jim Mims, our teacher, for all of his time and effort. Before this project, it would have been hard to handle such a large program and project. Now, with his help, we can do many of these actions with much greater speed and effectiveness. He has also taught us many ways in which one must approach these sorts of problems that has been helpful here and elsewhere.

We would also like to extend our thanks to Gene Wong, M.D. His invaluable time gave us the information we needed to write the program and do the project. Most of the sources we had only gave the basics and were intended for patients and their families. Dr. Wong's information, though, gave us what we needed to understand what is actually going on, an essential trait for this project.

CITATIONS

BOOKS

Deitel, H. M., Deitel, P. J., and Nieto, T. R. *Visual Basic 6: How to Program.* Upper Saddle River, NJ: Prentice-Hall, Inc., 1999.

Smith, Ella Thea, and Lorenzo Lisonbee. *YourBiology.* Second Edition. New York: Harcourt, Brace, and World Inc., 1962.

WEBSITES

American Academy of Dermatology. "Melanoma Fact Sheet", "Malignant Melanoma" <www.aad.org>

American Cancer Society. “All About Skin Cancer - Melanoma”, “What is Melanoma Skin Cancer?”, “Can Melanoma Be Prevented?”, “What Are the Risk Factors for Melanoma?” <www.cancer.org>

Aetna InteliHealth. “Melanoma”, “Skin Cancer:Basics”, “Cancer” <www.intelihealth.com >

Bosenberg, Marcus W. “Skin Cancer Models.” *National Cancer Institute: MMHCC The Mouse Models of Human Cancers Consortium.* 2003. <http://emice.nci.nih.gov/mouse_models/organ_models/skin_models >

Cancer Research Institute. “Conquering Melanoma: Prevent It, Spot It, Treat It” <http://www.cancerresearch.org/melanomabook.html>

Cotterill, Simon. “Medical Terminology and Cancer.” 2000. <http://www.cancerindex.org/medterm/>

Haggerty, Maureen. “Don’t Go For the Burn.” *Dermatology Insights.* Spring 2000: 11-13. <http://www.aad.org/NR/rdonlyres/933E3A27-8079-4425-9B56-387CE9D30909/0/DIspring00.pdf#page=12>

Melanoma…The ABC's. “What is Melanoma?”, “Melanoma Prevention”, “Treating Melanoma”, “Diagnosing Melanoma”,“Am I at Risk?” <www.melanoma.com>

National Center for Chronic Disease Prevention and Health Promotion. “Skin Cancer: Preventing America’s Most Common Cancer” <www.cdc.gov>

Orfield, Kevin. “Troubling but Treatable.” *Dermatology Insights.* Spring 2001: 9. <http://www.aad. org/NR/rdonlyres/0807BF94-1D74-4E85-A427-38ED646157EC/0/DIspring01.pdf#page=14>

Osterweil, Neil. “Want Melanoma? Get a Tan.” *WebMD Medical News*. 2004. <http://my.webmd.com/content/article/82/97076.htm>

Swanson, Jason R., and Jeffrey L. Melton, M.D. “Malignant Melanoma.” *Loyola University Medical Education Network: Skin Cancer and Benign Tumor Image Atlas.* 1996 <www.meddean.luc.edu/lumen/MedEd/medicine/dermatology/content.htm>

WebMD Health. “Health Guide A-Z: Skin Cancer, Melanoma” <http://my.webmd.com/hw/cancer/hw206550.asp>

DVDs

Microsoft Encarta Reference Library Premium 2005 DVD. “Skin Cancer.” Microsoft: 2004. DVD.

APPENDIX A: FIGURES FROM THE REPORT

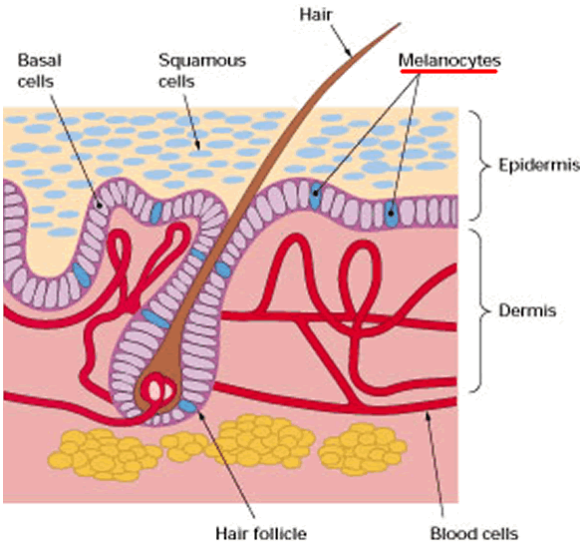

Figure 1: Cross-section of human skin.
< http://www.cancercouncil.com.au/editorial.asp?pageid=57 >

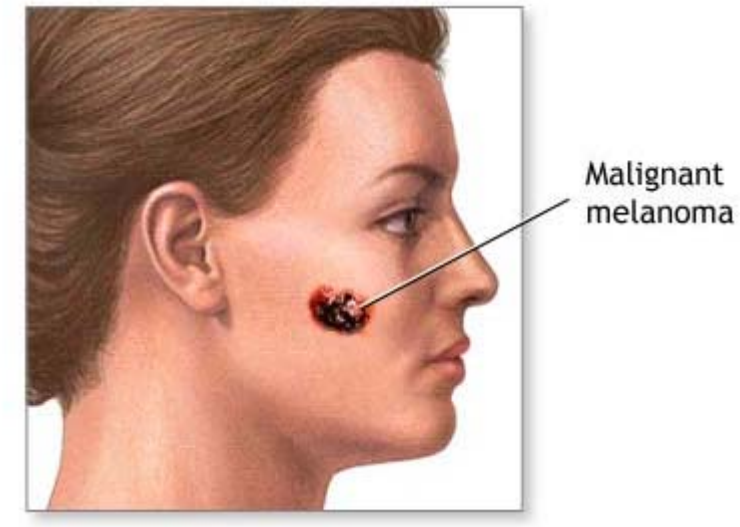

Figure 2: A graphical representation of malignant melanoma.
< http://www.nlm.nih.gov/medlineplus/ency/imagepages/9522.htm >

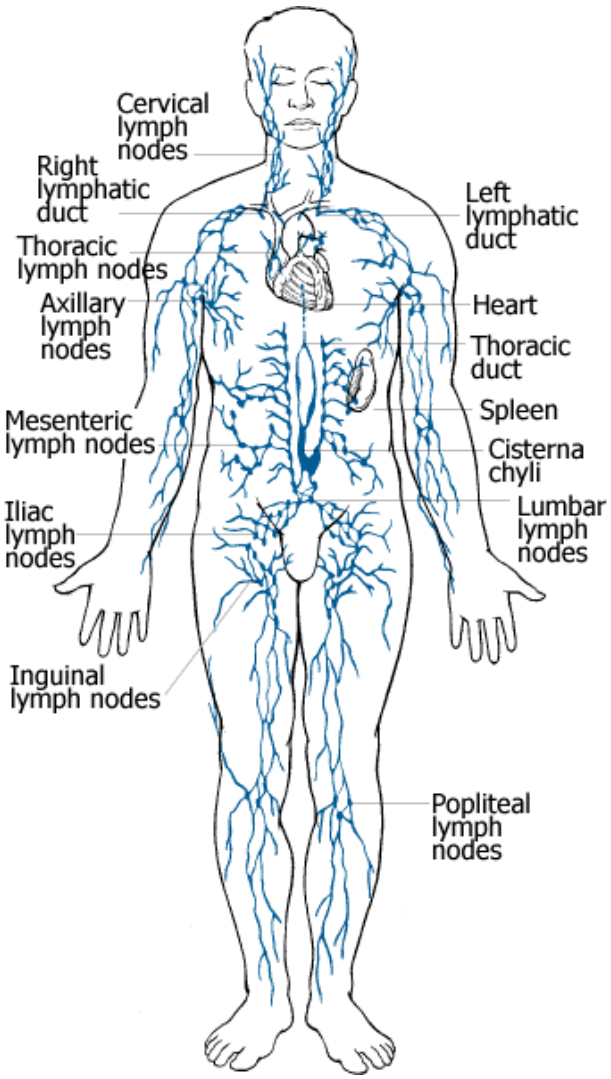

Figure 3: The human lymphatic system.
< http://www.gorhams.dk/assets/images/lymfesystemet.gif >

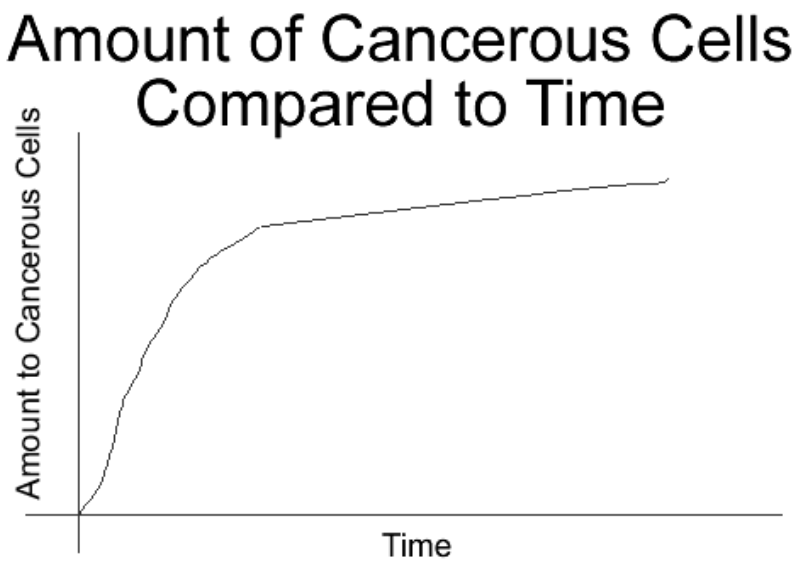

Figure 4: A generic sigmoidal curve. There is initially exponential growth, which then levels out as resource competition increases. Note this is not a program output.

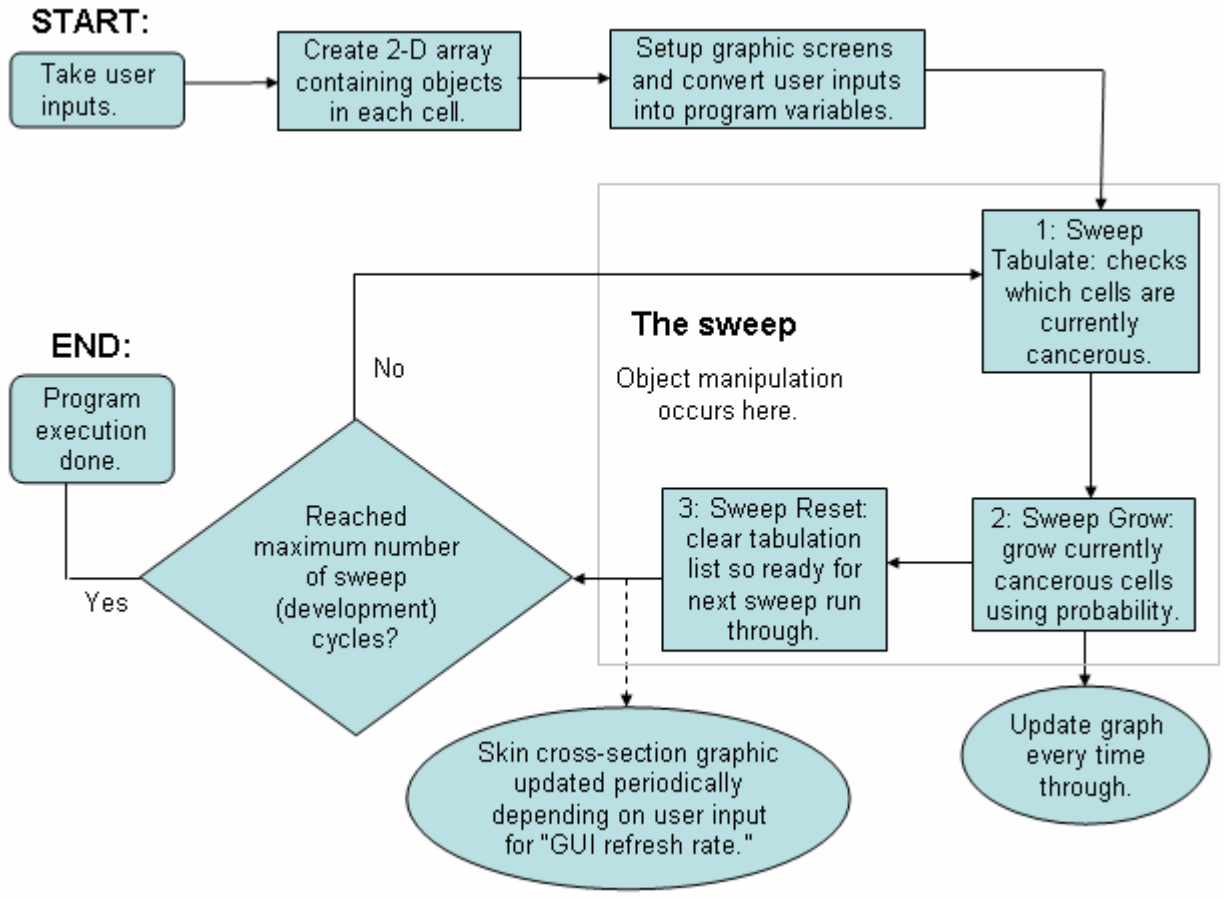

Figure 5: A simplified flowchart of the general algorithm used in the program. Note the sweep process in the grey box. This is where all of the cancer development is occurring.

APPENDIX B: APPLICATION SCREEN SHOTS

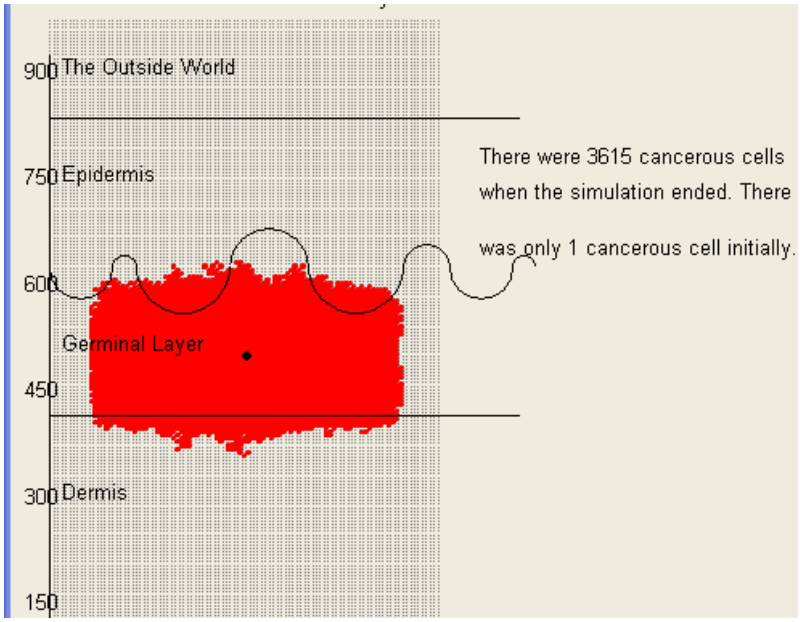

Figure 1: The elliptical cancer growth of a later version of our program.

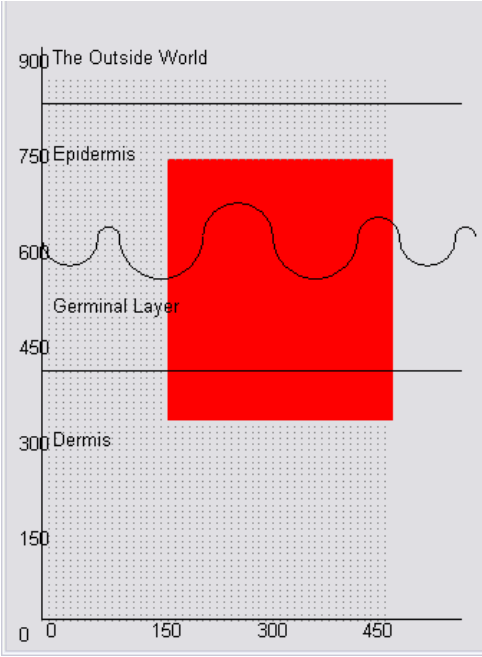

Figure 2: An earlier output of the program. Then, the growth was a square expanding from the point where the cancer originated.

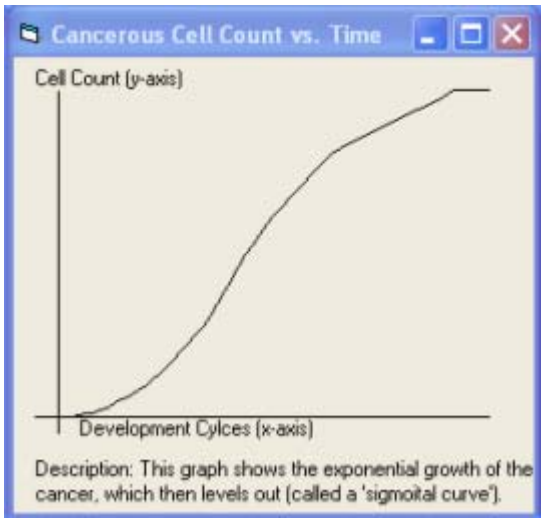

Figure 3: A screen capture of the graph screen from our program, demonstrating the sigmoidal curve.

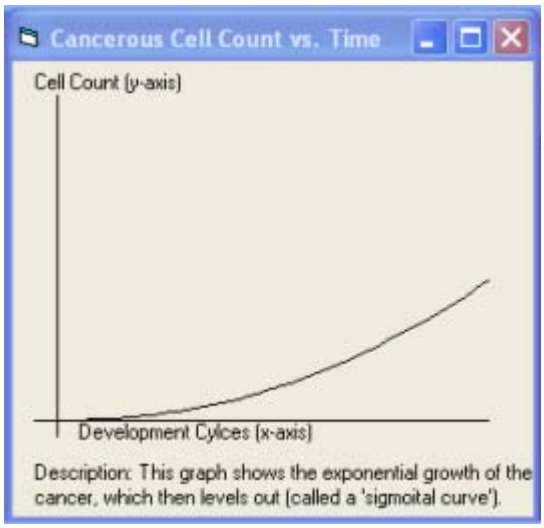

Figure 4: A screen capture of the growth as it is growing when there are almost no biological restrictions. Not the formation of an exponential curve.

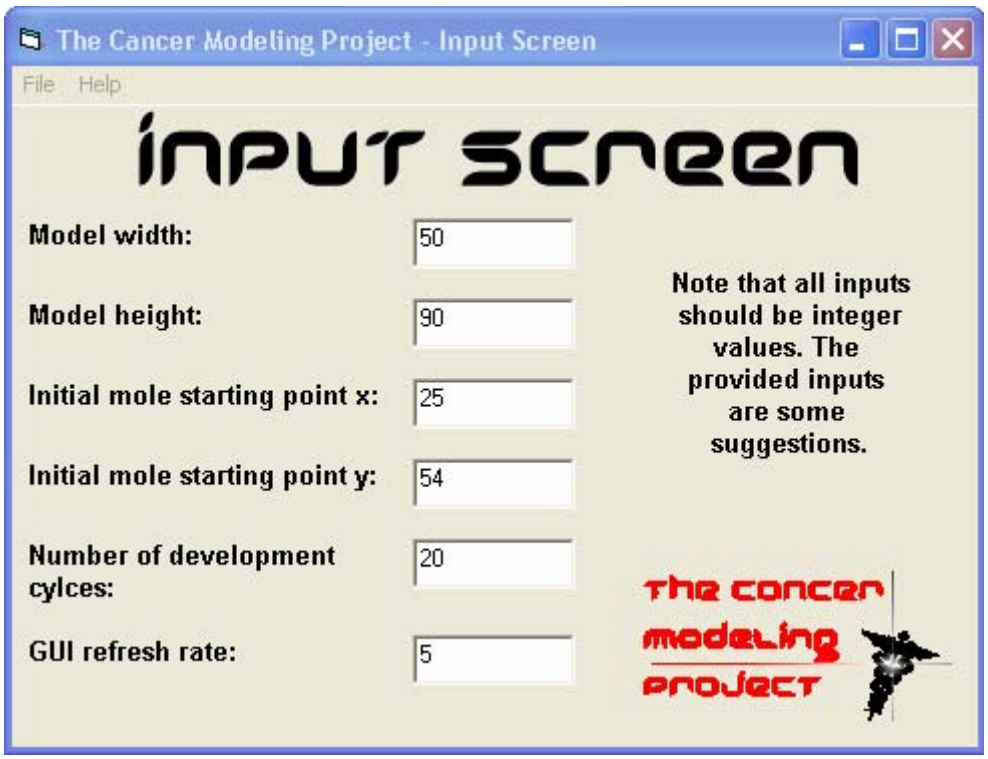

Figure 5: The input screen. From here, the user can specify the different user-input variables.

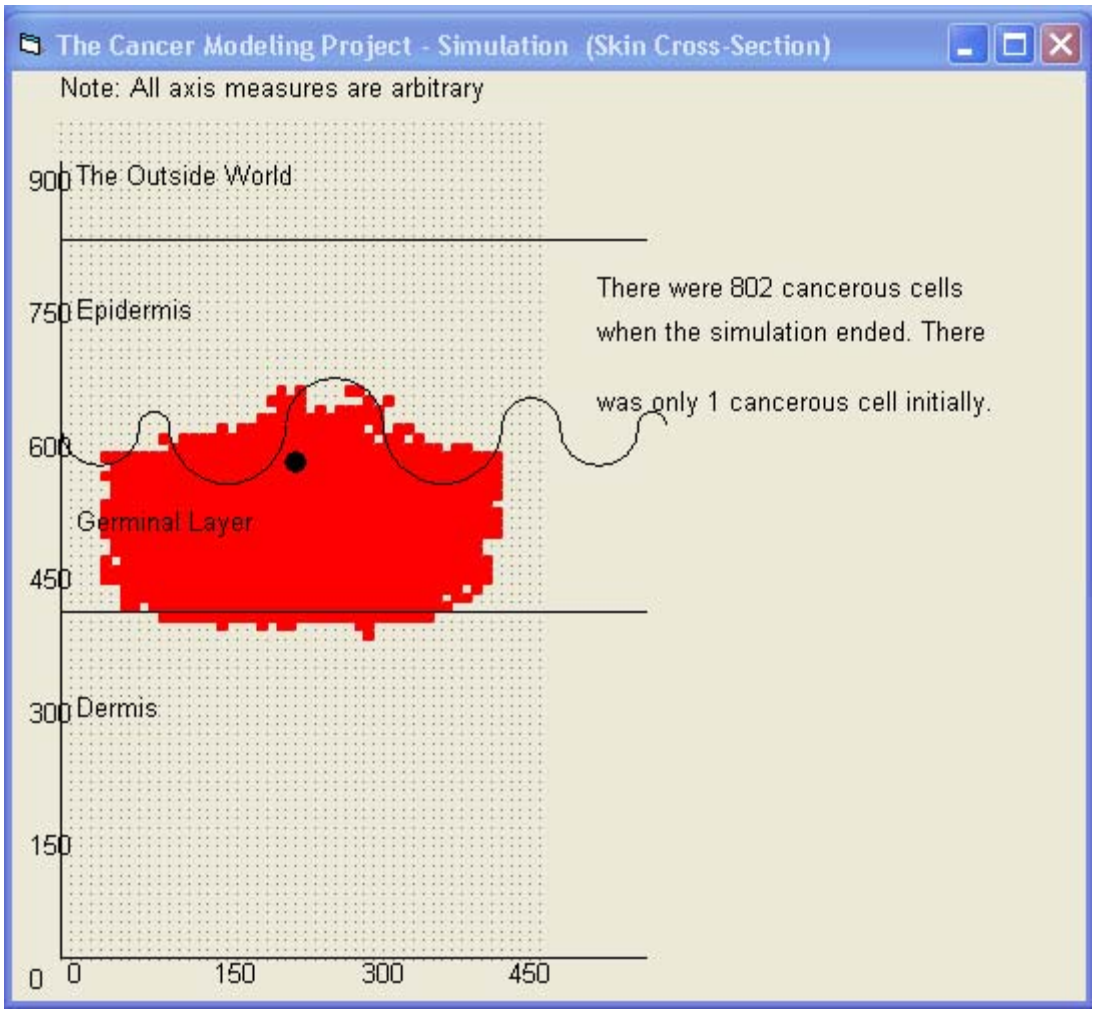

Figure 6: The cross-section representation of the skin with the red cancer developing in it.

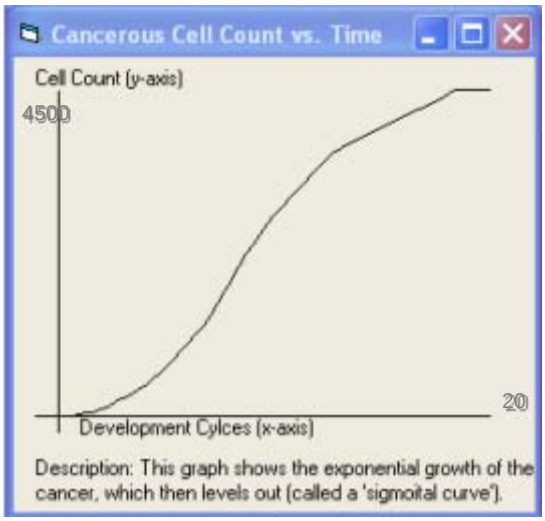

Figure 7: The output graph with the sigmoidal curve.

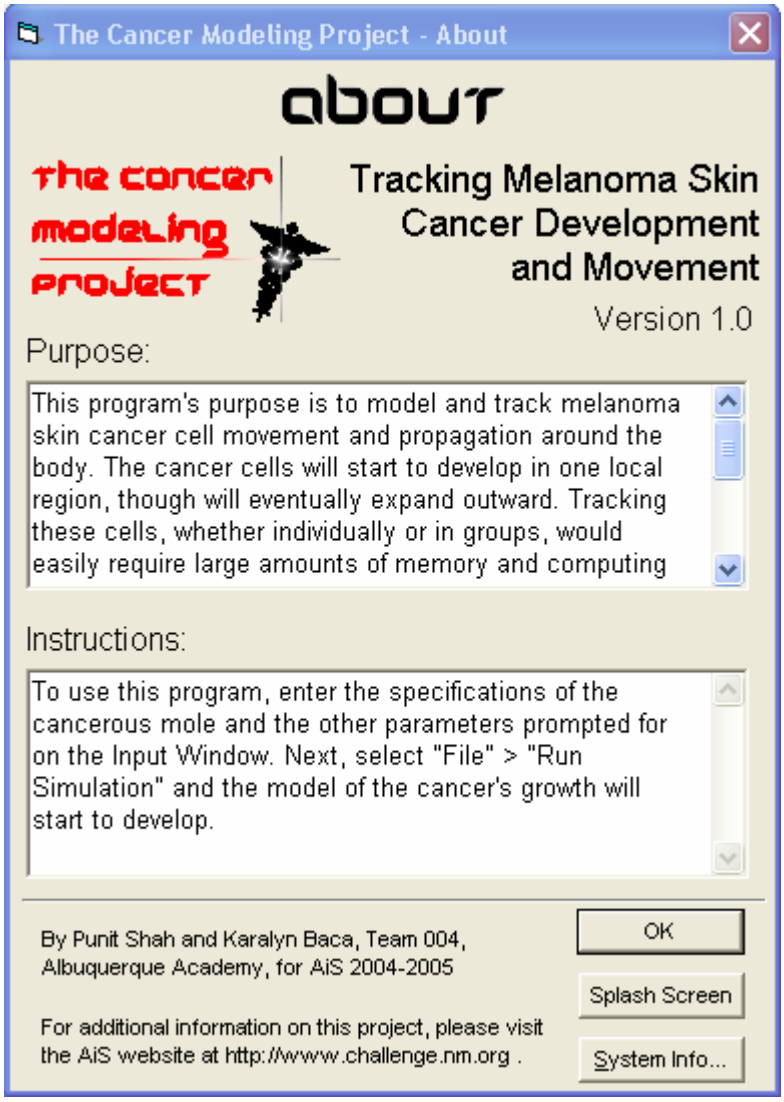

Figure 8: The help/about screen. The form describes the purpose of the project and instructions for usage of the program.

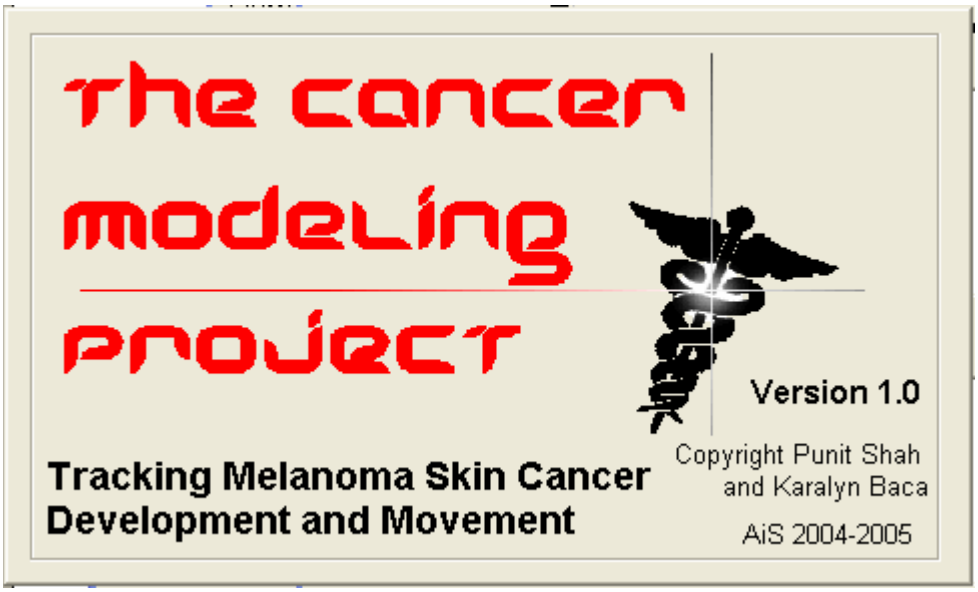

Figure 9: The splash screen.

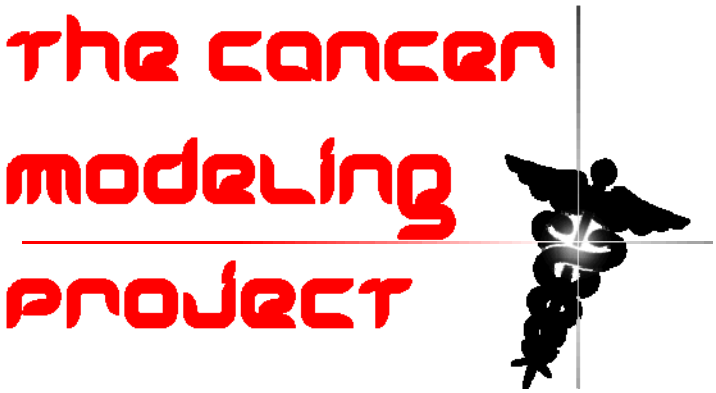

Figure 10: The project symbol, developed by us. Appears on several program screens.

APPENDIX C: FRMINPUT CODE

THE USER INPUT SCREEN

'Please note that there is a flowchart of the algorithms in this code in Appendix A, Figure 5

```
'The Cancer Modeling Project: Tracking Cancer Development and Movement
'Punit Shah and Karalyn Baca
'Team 004
'Albuquerque Academy
'AiS 2004-2005

'Code written in Microsoft Visual Basic 6.0, line comment charecter is '

'Code for: frmInput
'Purpose of module: Where the data is inputed and then all actions are delt with

Option Explicit 'Force variable declerations

'Dim Modular vars (not project-wide visible)
Private ModelSizeX As Integer   'The height of the model
Private ModelSizeY As Integer   'The width of the model
Private CancerModel() As GraphicCell 'Defines the array which will hold the objects of the
GraphicsCell class
Private SweepCounter As Integer

'COMMENTS FOR OUR SELVES:
'-When we finish the program, make frmSplash the starting screen

'This is the main procedure that is called when the user clicks the "Run Simulation" button
Private Sub mnuMain_Click()
   'Purpose: to initiate all actions to be taken, project main

   'Dim vars that then take user inputs and set themselves equal to the appropriate one
   Dim StartX As Integer   'Mole starting position, x coordinate part
   StartX = txtStartX.Text - 1
   Dim StartY As Integer   'Mole starting position, y coordingate part
   StartY = txtStartY.Text - 1
   Dim GUIRefreshRate As Integer   'How often program should update the primary model GUI
   GUIRefreshRate = txtGUIRefresh.Text
   Dim DevCycles As Integer        'How many sweep-sets we should do to develop the cacner
   DevCycles = txtDevCycles.Text

   ModelSizeX = txtModelSizeX.Text - 1     'Sets the model width to what it should be
   ModelSizeY = txtModelSizeY.Text - 1     'Sets the model height to what it should be

   'Give a size to the array organizing all of the objects of the model
```

```
    'Have to do this here instead of above as VB only allows for this assignment to occur
    'inside of a procedure
    ReDim CancerModel(ModelSizeX + 1, ModelSizeY + 1) As GraphicCell

    'Do the modeling
    Call InitiateAction(StartX, StartY, DevCycles, GUIRefreshRate)

    'Call all closing procedures
    Call FinishingActions

End Sub

Private Sub InitiateAction(ByRef MoleStartingX As Integer, ByRef MoleStartingY As Integer,
ByRef TotalDevCycles As Integer, ByRef GUIRefreshRate As Integer)
    'Purpose: to initiate following the model until the time when the user is done modeling

    'For loop Variable Declarations
    Dim x As Integer
    Dim y As Integer

    'Prepare the graph form for the graph
    Call SetUpGraph(TotalDevCycles)


    'Now create the objects of the class for each cell of the graphics
    For x = 0 To ModelSizeX
        For y = 0 To ModelSizeY
            Set CancerModel(x, y) = New GraphicCell
        Next y
    Next x

    'Create the initial cancer at location specified by user
    CancerModel(MoleStartingX, MoleStartingY).IsCancerous = True

    'Show Simulation Screens
    frmSimulation.Show  'Where the simulation is draws
    frmCellCount.Show   'Where the cell count graph will go

    'For loop to have development cylces go through specified amount of times
    For SweepCounter = 1 To TotalDevCycles

        'Go throught the sweeps
        Call SweepTabulate  'Sweep #1, Tabulate, see procedure for purpose
        Call SweepGrow      'Sweep #2, Grow, see procedure for purpose
        Call SweepReset     'Sweep #3, Reset, see procedure for purpose
```

```
        'Check to see if we need to update the cross section of the skin
        If SweepCounter Mod GUIRefreshRate = 0 Then

            Call PrepForm   'Clears screen and scales for easier drawing

            'Print the cancerous cell data to the screen
            For x = 0 To ModelSizeX     'loop through "x"s
               For y = 0 To ModelSizeY      'loop through "y"s

                  If CancerModel(x, y).IsCancerous = True Then

                     frmSimulation.Line (x, y)-(x + 1, y + 1), vbRed, BF     'Red cancer cells (squares)

                  Else

                     frmSimulation.PSet (x, y), QBColor(8) ' Grey health cell (dots)

                  End If

               Next y
            Next x

            'Draws small circle for the starting point
            frmSimulation.FillStyle = 0
            frmSimulation.Circle (MoleStartingX, MoleStartingY), 1, vbBlack

            'Draw the skin layers and model axises ontop of the cancer model
            frmSimulation.Scale (-50, 1000)-(1050, -50) 'Changed scale so that over-diagram looks
right
            Call DrawAxes
            Call DrawLayers

        End If

    Next SweepCounter


End Sub

Private Sub PrepForm()
    'Purpose: to prepare the form for the cancer model cross-section

    'clear form
    frmSimulation.Cls

    'scale form for easier numbers to draw with
```

```
   frmSimulation.Scale (-(1 / 20) * (2 * (ModelSizeX + 1)), (21 / 20) * (ModelSizeY + 1))-((21 /
20) * (2 * (ModelSizeX + 1)), -(1 / 20) * (ModelSizeY + 1))

End Sub

Private Sub DrawAxes()
   'Purpose: to draw x and y axis of the frmSimulation form

   'x-axis
   frmSimulation.Line (0, 0)-(600, 0)

   'y-axis
   frmSimulation.Line (0, 0)-(0, 900)

   'label the x-axis
   Dim x As Integer
   For x = 0 To 450 Step 150
         frmSimulation.CurrentX = x
         frmSimulation.CurrentY = 0
         frmSimulation.Print x
   Next x

   'label the y-axis
   Dim i As Integer
   For i = 0 To 900 Step 150
      frmSimulation.CurrentX = -39
      frmSimulation.CurrentY = i - 6
      frmSimulation.Print i
   Next i

End Sub

Private Sub DrawLayers()
   'Purpopse: to draw the layers of the skin onto the model

   'Make and set Pi
   Dim Pi As Double
   Pi = 3.14159265357932

   'Sets font/size for the form
   frmSimulation.Font = "Arial"
   frmSimulation.FontSize = 10


   'Below are the lines of code that write the different names of the layers
   'and also draw the strait/wavy lines that define these layers
```

```
    frmSimulation.CurrentX = 15
    frmSimulation.CurrentY = 900
    frmSimulation.Print "The Outside World"


    frmSimulation.Line (0, 810)-(600, 810) 'top of epidermis
    frmSimulation.CurrentX = 15
    frmSimulation.CurrentY = 750
    frmSimulation.Print "Epidermis"

    'To make the wavy lines for the layer
    frmSimulation.Circle (40, 600), 40, , Pi, 2 * Pi
    frmSimulation.Circle (95, 600), 15, , 0, Pi
    frmSimulation.Circle (170, 600), 60, , Pi, 2 * Pi
    frmSimulation.Circle (280, 600), 50, , 0, Pi
    frmSimulation.Circle (390, 600), 60, , Pi, 2 * Pi 'Wavy top of germanal
    frmSimulation.Circle (480, 600), 30, , 0, Pi
    frmSimulation.Circle (550, 600), 40, , Pi, 2 * Pi
    frmSimulation.Circle (605, 600), 15, , 0, Pi
    frmSimulation.CurrentX = 15
    frmSimulation.CurrentY = 510
    frmSimulation.Print "Germinal Layer"

    frmSimulation.Line (0, 390)-(600, 390)
    frmSimulation.CurrentX = 15
    frmSimulation.CurrentY = 300
    frmSimulation.Print "Dermis"

    'Note for the user that measurements along y-axis are arbitrary
    frmSimulation.CurrentX = 0
    frmSimulation.CurrentY = 1000
    frmSimulation.Print "Note: All axis measures are arbitrary"

End Sub

Private Sub SweepTabulate()
    'Purpose: this is the first sweep. It checks which cells are cancerous and notes those which
currently
          ' are. This is so the cells that get cancerous during this development cycle are not further
developed
          ' during this development cycle.

    'Variable definitions

    'Counters
```

```
    Dim Counter1 As Integer, Counter2 As Integer

    'We will use this information for the graph (counts cancerous cells)
    Dim CellCounter As Integer

    'Sweep through and check for cancerous cells and notate those cells
    For Counter1 = 0 To ModelSizeX  'goes through "x"s
        For Counter2 = 0 To ModelSizeY  'goes through "y"s

            'Checks to see if the current cell is cancerous
            If CancerModel(Counter1, Counter2).IsCancerous = True Then

                'If cell is cancerous, notates in "HasBeenSwept" property
                CancerModel(Counter1, Counter2).HasBeenSwept = True

                'Adds to the counter of how many cancerous cells
                CellCounter = CellCounter + 1

            End If

        Next Counter2
    Next Counter1

    'Do the gaphing of amount of cells, passing how many cancerous cells
    Call UpdateGraph(CellCounter)

End Sub

Private Sub SweepGrow()
    'Purpose: this is the second sweep. It develops the cells that were deemed cancerous in the
            ' previous sweep.

    'Variable definitions

    'Counters
    Dim Counter1 As Integer, Counter2 As Integer

    'Goes through model and calls procedure to check and infect neighbors
    For Counter1 = 0 To ModelSizeX
        For Counter2 = 0 To ModelSizeY

        'Checks if this was one of the previously notated cells as notated in sweep 1
        If CancerModel(Counter1, Counter2).HasBeenSwept = True Then

            'Calls sub to infect neighbors of this cancerous cell
            Call InfectNeighbors(Counter1, Counter2)
```

```
        End If

        Next Counter2
    Next Counter1

End Sub

Private Sub SweepReset()
    'Purpose: this is sweep three. Here, the program changes all of the HasBeenSwept properties
          ' to false so that next time we do sweep 1, we will have a clean "HasBeenSwept" property
          ' to work with.

    'Variable definitions

    'Counters
    Dim Counter1 As Integer, Counter2 As Integer

    'Sweeps through the model and resets "HassBeenSwep property
    For Counter1 = 0 To ModelSizeX
        For Counter2 = 0 To ModelSizeY

        'If perviously notated cell, only then resets
        If CancerModel(Counter1, Counter2).HasBeenSwept = True Then

            'Sets property to original value
            CancerModel(Counter1, Counter2).HasBeenSwept = False

        End If

        Next Counter2
    Next Counter1

End Sub

Private Sub InfectNeighbors(ByVal LocationX As Integer, ByVal LocationY As Integer)
    'Purpose: to check to see if neighbors can be infected and then infects if can be

    'variables to use to check new coordinates
    Dim CurrentNeighborX As Integer
    Dim CurrentNeighborY As Integer


    'Goes through the different 8 neighbors checking

    'NW location
```

```
    CurrentNeighborX = LocationX - 1
    CurrentNeighborY = LocationY + 1

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If


    'N location
    CurrentNeighborX = LocationX
    CurrentNeighborY = LocationY + 1

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If


    'NE location
    CurrentNeighborX = LocationX + 1
    CurrentNeighborY = LocationY + 1

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If
```

```
    'E location
    CurrentNeighborX = LocationX + 1
    CurrentNeighborY = LocationY

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If


    'SE location
    CurrentNeighborX = LocationX + 1
    CurrentNeighborY = LocationY - 1

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If


    'S location
    CurrentNeighborX = LocationX
    CurrentNeighborY = LocationY - 1

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
```

```
    End If


    'SW location
    CurrentNeighborX = LocationX - 1
    CurrentNeighborY = LocationY - 1

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If


    'W location
    CurrentNeighborX = LocationX - 1
    CurrentNeighborY = LocationY

    'Check to see if the points are in range, and if so develop the cancer in this direction
    If CurrentNeighborX > -1 And CurrentNeighborX <= ModelSizeX And CurrentNeighborY > -
1 And CurrentNeighborY <= ModelSizeY Then
        If ShouldIInfect(CurrentNeighborX, CurrentNeighborY) = True Then

            'Check to see probabiliy of infection due to location and then infect if probability correct
            CancerModel(CurrentNeighborX, CurrentNeighborY).IsCancerous = True

        End If
    End If


End Sub

Private Function ShouldIInfect(ByVal LocationX As Integer, ByVal LocationY As Integer) As
Boolean
    'Purpose: This is where many of the factors of the program come together to determine
            'if the cell around it should be infected taking into effect probability calculated
            'using infromation about the different influencing factors.

    Randomize   'Makes the random numbers truly random instead of off of a list for testing
```

```
    'Variable definitions

    'The random number determind later in the sub
    Dim RandomNumber As Integer

    'Defines counter for adding odds and sets to 1 (1/1 odds initially)
    Dim ProbabilityCounter As Integer
    ProbabilityCounter = 1

    'These following statements will add to the counter which will be used for the probablility

    'Skin layer preassure and other primary physical infuelces
    If (1 / 10) * ModelSizeY < Abs((ModelSizeY / 2) - LocationY) Then
        ProbabilityCounter = ProbabilityCounter + 15
    End If

    ProbabilityCounter = ProbabilityCounter + 1     'Natural cell growth chance


    'Use the counter to determine whether the cancer should grow at current location

    RandomNumber = Int(ProbabilityCounter * Rnd)

    If RandomNumber = 0 Then
        ShouldIInfect = True

    Else
        ShouldIInfect = False

    End If

End Function

Private Sub SetUpGraph(ByVal MaxDevCycles As Integer)
    'Purpose: to set up the graph form for taking the graph inputs

    frmCellCount.Cls    'Earases form, gives us clean board to draw on

    Dim MaxCellCount As Double
    MaxCellCount = (ModelSizeX + 1) * (ModelSizeY + 1)

    'Scales the form so numbers work out easier when drawing to form
    frmCellCount.Scale (-(3 / 20) * MaxDevCycles, (22 / 20) * MaxCellCount)-((22 / 20) *
MaxDevCycles, -(6 / 20) * MaxCellCount)

    'x-axis label
```

```
    frmCellCount.CurrentX = (1 / 20) * MaxDevCycles
    frmCellCount.CurrentY = 0
    frmCellCount.Print "Development Cylces (x-axis)"

    'x-axis max value
    frmCellCount.CurrentX = (20.5 / 20) * MaxDevCycles
    frmCellCount.CurrentY = 0
    frmCellCount.Print MaxDevCycles

    'y-axis label
    frmCellCount.CurrentX = -(1 / 20) * MaxDevCycles
    frmCellCount.CurrentY = (21.5 / 20) * MaxCellCount
    frmCellCount.Print "Cell Count (y-axis)"

    'y-axis max value
    frmCellCount.CurrentX = -(3 / 20) * MaxDevCycles
    frmCellCount.CurrentY = MaxCellCount
    frmCellCount.Print MaxCellCount

    'Draw axises
    frmCellCount.Line (-(1 / 20) * MaxDevCycles, 0)-(MaxDevCycles, 0) 'x-axis
    frmCellCount.Line (0, -(1 / 20) * MaxCellCount)-(0, MaxCellCount) 'y-axis

    'Write Description
    frmCellCount.CurrentX = -(1 / 20) * MaxDevCycles
    frmCellCount.CurrentY = MaxCellCount * -(2.5 / 20)
    frmCellCount.Print "Description: This graph shows the exponential growth of the"
    frmCellCount.CurrentX = -(1 / 20) * MaxDevCycles
    frmCellCount.CurrentY = MaxCellCount * -(4 / 20)
    frmCellCount.Print "cancer, which then levels out (called a 'sigmoital curve')."

End Sub

Private Sub UpdateGraph(ByVal CurrentCellCount As Integer)
    'Purpose: to update the graph of how many cancerous cells there are

    'Variable definitions

    'This is the var to track where the previous point was so we know where to
    'draw the new line of the graph from
    'Don't have to track the old x-coordinate as this is always sweepcounter - 1

    'Previous number of cells, set initially to 0 (for firts coordinate)
    Static PreviousY As Integer
    If SweepCounter = 1 Then
        PreviousY = 0
```

```
    End If

    'Graph the new point by drawing line from previous one to new one
    frmCellCount.Line (SweepCounter - 1, PreviousY)-(SweepCounter, CurrentCellCount)

    'Make the current points the now-previous points
    PreviousY = CurrentCellCount

End Sub


Private Sub FinishingActions()
    'Purpose: to wrap up the program and do some ending commands

    'Make "beep" and come up with a message box notifying user program is done modeling to
point specified
    Beep
    MsgBox ("The propgram has finished modeling the cancer growth to the stage of development
you chose to end at. Thank you for using The Cancer Modeling Project. Feel free to enter new
values and re-run.") ', , "The Cancer Modeling Project - Simulation Completed")

    'Counts the number of cancerous cells
    Dim x As Integer, y As Integer
    Dim CancerCellCount As Integer

    For x = 0 To ModelSizeX
        For y = 0 To ModelSizeY

            If CancerModel(x, y).IsCancerous = True Then

                'Adds to counter if cancerous
                CancerCellCount = CancerCellCount + 1

            End If

        Next y
    Next x

    'Prints the message of how many cancerous cells there were when the simulation was done
running
    frmSimulation.CurrentX = 550
    frmSimulation.CurrentY = 775
    frmSimulation.Print "There were " & CancerCellCount & " cancerous cells"
    frmSimulation.CurrentX = 550
    frmSimulation.CurrentY = 725
    frmSimulation.Print "when the simulation ended. There"
```

```
    frmSimulation.CurrentX = 550
    frmSimulation.CurrentY = 645
    frmSimulation.Print "was only 1 cancerous cell initially."

End Sub

'Makes whole program end when the input screen is closed
Private Sub Form_Unload(Cancel As Integer)
    End
End Sub

'Other menu bottons procedures that are activated when they are clicked

Private Sub mnuExit_Click()
    End 'End program
End Sub

Private Sub mnuAbout_Click()
    frmAbout.Show   'Show the About screen
End Sub
```

APPENDIX D: GRAPHICCELL CODE

THE CLASS OF ALL OF THE MODEL'S CELLS

```
'The Cancer Modeling Project: Tracking Cancer Development and Movement
'Punit Shah and Karalyn Baca
'Team 004
'Albuquerque Academy
'AiS 2004-2005

'Code written in Microsoft Visual Basic 6.0, line comment character is '

'Code for: GraphicCell
'Purpose of class module: To be the class in which all of the cells of the graphics become objects
of.

Option Explicit 'Force variable declarations

'Public Variables

'The variable that determines whether a certain cell is cancerous or not
Public IsCancerous As Boolean

'Variable used for tabulation in sweeps when developing cancer (see frmInput code)
Public HasBeenSwept As Boolean

'Each number is a loctaion and it tell whether that neighbor is cancerous
Public Neighbors(8) As Boolean


Public Function Nutrients() As Double
   'Purpose: To calculate the number of neighbors with cancer, and then using that, calculate the
amount of avalible nutrients
         ' This is very important as it is a factor in cancer growth
   Dim Counter As Integer
   Dim HowManyNeighbors As Integer

   For Counter = 0 To 7
      If Neighbors(Counter) = True Then
         HowManyNeighbors = HowManyNeighbors + 1
      End If
   Next Counter

   Nutrients = 100 - HowManyNeighbors * (100 / 8)
End Function
```

APPENDIX E: FRMABOUT CODE

THE APPLICATION INFORMATION SCREEN

```
'The Cancer Modeling Project: Tracking Cancer Development and Movement
'Punit Shah and Karalyn Baca
'Team 004
'Albuquerque Academy
'AiS 2004-2005
'Parts of code in this section (frmAbout) were generated automatically by VB

'Code written in Microsoft Visual Basic 6.0, line comment charecter is '

'Code for: frmAbout
'Purpose of module: Show program's reason and instructions for use

Option Explicit 'Force variable declerations

' Reg Key Security Options...
Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE + _
                KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS + _
                KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL

' Reg Key ROOT Types...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1                        ' Unicode nul terminated string
Const REG_DWORD = 4                     ' 32-bit number

Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"

Private Declare Function RegOpenKeyEx Lib "advapi32" Alias "RegOpenKeyExA" (ByVal
hKey As Long, ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As
Long, ByRef phkResult As Long) As Long
```

```
Private Declare Function RegQueryValueEx Lib "advapi32" Alias "RegQueryValueExA"
(ByVal hKey As Long, ByVal lpValueName As String, ByVal lpReserved As Long, ByRef
lpType As Long, ByVal lpData As String, ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) As Long


Private Sub cmdSysInfo_Click()
  Call StartSysInfo
End Sub

Private Sub cmdOK_Click()
  Unload Me
End Sub

Private Sub cmdSplashShow_Click()
   frmSplash.Show
End Sub

Private Sub Form_Load()
   'Me.Caption = "About " & App.Title
   'lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
   'lblTitle.Caption = App.Title
End Sub

Public Sub StartSysInfo()
   On Error GoTo SysInfoErr

   Dim rc As Long
   Dim SysInfoPath As String

   ' Try To Get System Info Program Path\Name From Registry...
   If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO, gREGVALSYSINFO,
SysInfoPath) Then
   ' Try To Get System Info Program Path Only From Registry...
   ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC,
gREGVALSYSINFOLOC, SysInfoPath) Then
      ' Validate Existance Of Known 32 Bit File Version
      If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
         SysInfoPath = SysInfoPath & "\MSINFO32.EXE"

      ' Error - File Can Not Be Found...
      Else
         GoTo SysInfoErr
      End If
   ' Error - Registry Entry Can Not Be Found...
   Else
```

```
    GoTo SysInfoErr
  End If

  Call Shell(SysInfoPath, vbNormalFocus)

  Exit Sub
SysInfoErr:
  MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub

Public Function GetKeyValue(KeyRoot As Long, KeyName As String, SubKeyRef As String,
ByRef KeyVal As String) As Boolean
  Dim i As Long                                ' Loop Counter
  Dim rc As Long                               ' Return Code
  Dim hKey As Long                             ' Handle To An Open Registry Key
  Dim hDepth As Long                           '
  Dim KeyValType As Long                       ' Data Type Of A Registry Key
  Dim tmpVal As String                         ' Tempory Storage For A Registry Key Value
  Dim KeyValSize As Long                       ' Size Of Registry Key Variable
  '------------------------------------------------------------
  ' Open RegKey Under KeyRoot {HKEY_LOCAL_MACHINE...}
  '------------------------------------------------------------
  rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) ' Open Registry
Key

  If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError         ' Handle Error...

  tmpVal = String$(1024, 0)                    ' Allocate Variable Space
  KeyValSize = 1024                            ' Mark Variable Size

  '------------------------------------------------------------
  ' Retrieve Registry Key Value...
  '------------------------------------------------------------
  rc = RegQueryValueEx(hKey, SubKeyRef, 0, _
                  KeyValType, tmpVal, KeyValSize)    ' Get/Create Key Value

  If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError         ' Handle Errors

  If (Asc(Mid(tmpVal, KeyValSize, 1)) = 0) Then          ' Win95 Adds Null Terminated String...
    tmpVal = Left(tmpVal, KeyValSize - 1)              ' Null Found, Extract From String
  Else                                         ' WinNT Does NOT Null Terminate String...
    tmpVal = Left(tmpVal, KeyValSize)                ' Null Not Found, Extract String Only
  End If
  '------------------------------------------------------------
  ' Determine Key Value Type For Conversion...
  '------------------------------------------------------------
```

```
   Select Case KeyValType                          ' Search Data Types...
   Case REG_SZ                                    ' String Registry Key Data Type
      KeyVal = tmpVal                             ' Copy String Value
   Case REG_DWORD                                    ' Double Word Registry Key Data Type
      For i = Len(tmpVal) To 1 Step -1                ' Convert Each Bit
         KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1)))   ' Build Value Char. By Char.
      Next
      KeyVal = Format$("&h" + KeyVal)                 ' Convert Double Word To String
   End Select

   GetKeyValue = True                              ' Return Success
   rc = RegCloseKey(hKey)                            ' Close Registry Key
   Exit Function                                  ' Exit

GetKeyError:      ' Cleanup After An Error Has Occured...
   KeyVal = ""                                    ' Set Return Val To Empty String
   GetKeyValue = False                               ' Return Failure
   rc = RegCloseKey(hKey)                            ' Close Registry Key
End Function
```

APPENDIX F: FRMSIMULATION CODE

THE SKIN CROSS-SECTION: THE PROGRAM'S PRIMARY OUTPUT

```
'The Cancer Modeling Project: Tracking Cancer Development and Movement
'Punit Shah and Karalyn Baca
'Team 004
'Albuquerque Academy
'AiS 2004-2005

'Code written in Microsoft Visual Basic 6.0, line comment charecter is '

'Code for: frmSimulation
'Purpose of module: Show program's primary GUI output (the skin model)

Private Sub Form_Unload(Cancel As Integer)
   'Hides this window when the graph is closed, just for convinience
   frmCellCount.Hide
End Sub
```

APPENDIX G: FRMCELLCOUNT CODE

THE GRAPH OF THE NUMBER OF CELLS TO DEVELOPMENT CYCLES

```
'The Cancer Modeling Project: Tracking Cancer Development and Movement
'Punit Shah and Karalyn Baca
'Team 004
'Albuquerque Academy
'AiS 2004-2005

'Code written in Microsoft Visual Basic 6.0, line comment charecter is '

'Code for: frmCellCount
'Purpose of module: Show program's cancerous cell vs. time graph

Private Sub Form_Unload(Cancel As Integer)
    'Hides this window when the skin model is closed, just for convinience
    frmSimulation.Hide
End Sub
```

APPENDIX H: FRMSPLASH CODE
THE SPLASH SCREEN

```
'The Cancer Modeling Project: Tracking Cancer Development and Movement
'Punit Shah and Karalyn Baca
'Team 004
'Albuquerque Academy
'AiS 2004-2005
'Parts of code in this section (frmSplahs) were generated automatically by VB

'Code written in Microsoft Visual Basic 6.0, line comment charecter is '

'Code for: frmSplash
'Purpose of module: Splash Screen

Option Explicit 'Force variable declerations

Private Sub Form_KeyPress(KeyAscii As Integer)
   Call HideTheForm
End Sub

Private Sub Frame_Click()
   Call HideTheForm
End Sub

Private Sub imgLogo_Click()
   Call HideTheForm
End Sub

Private Sub tmrHideSplash_Timer()
   Call HideTheForm
End Sub

Private Sub HideTheForm()
   Static HiddenOnce As Boolean

   Unload Me
   tmrHideSplash.Enabled = False

   If HiddenOnce = False Then
      frmInput.Show
   End If

   HiddenOnce = True
End Sub
```