Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина» (УрФУ) Институт радиоэлектроники и информационных технологий – РтФ

Школа профессионального и академического образования

Учебно-научный центр «Информационная безопасность»

Оценка \_\_\_\_\_\_\_\_\_\_

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА ПО КУРСОВОЙ РАБОТЕ**

**«СПАСЕНИЕ МОРЯКА С ПОМОЩЬЮ ТЕОРЕМЫ БАЙЕСА»**

**ПО ДИСЦИПЛИНЕ: «ЯЗЫКИ И МЕТОДЫ ПРОГРАММИРОВАНИЯ»**

|  |  |
| --- | --- |
| Преподаватель | Мирвода С.Г. |
| Студенты | Ефремов Г.Е., Шабунин И.С., Шарова Д.А.. |
| Специальность | Информационная безопасность телекоммуникационных систем |
| Группа | РИ-311050 |

Екатеринбург

2024**ВВЕДЕНИЕ**

**1 Цели проекта**

Создание игры на языке Python, которая использует теорему Байеса для поиска одинокого моряка, пропавшего у мыса. Программа поможет спасателям (игрокам) эффективнее и быстрее найти и спасти пропавшего моряка.

**2 Видение проекта**

Проект «Спасение моряка с помощью теоремы Байеса» представляет собой игру на языке Python, разработанную с целью оптимизации поиска пропавшего в море моряка. С помощью теоремы Байеса игра рассчитывает вероятности нахождения моряка в зонах поиска и обновляет их по мере получения результатов поиска.

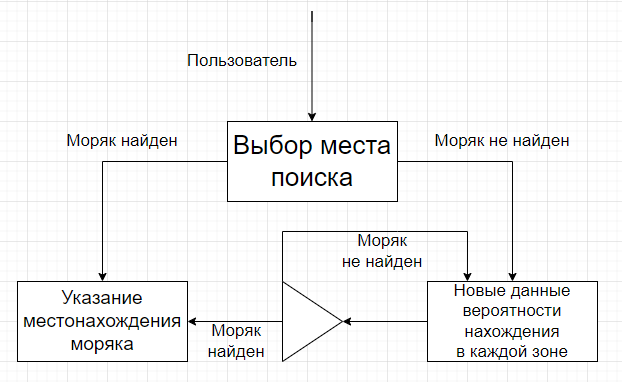


Рис. 1. Функциональная диаграмма видения продукта

Программа отображает карту, произвольно выбирает местоположение моряка, выводит меню вариантов поиска (с вариантами выбора области поиска) и либо показывает моряка при успешном нахождении, либо выполняет байесовский вывод для вероятностей нахождения в каждой области.

**БЛОК ПРОЕКТИРОВАНИЯ**

**1 Архитектура продукта**

В проекте не была реализована архитектура, так как наша программа представляет собой простое клиентское приложение

Архитектурная диаграмма

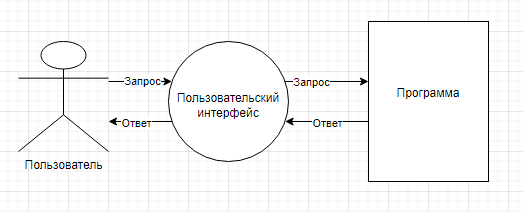


Рис. 2. Архитектурная диаграмма

**2 Требования к системному ПО**

Требования к ПО:

1. Windows 8, 10
2. Python 3.2 и выше
3. Скачанные библиотеки numpy, Matplotlib и OpenCV

При запуске программа автоматически начнет игру.

Функциональная диаграмма

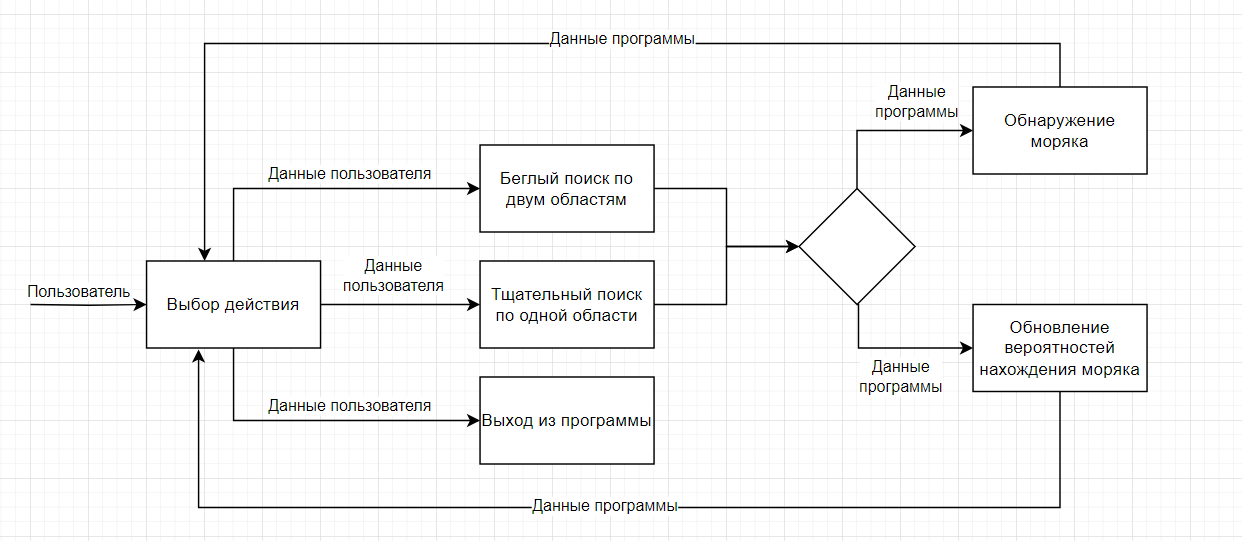


Рис. 3. Функциональная диаграмма

Диаграмма последовательности (функция добавление задачи в проект)

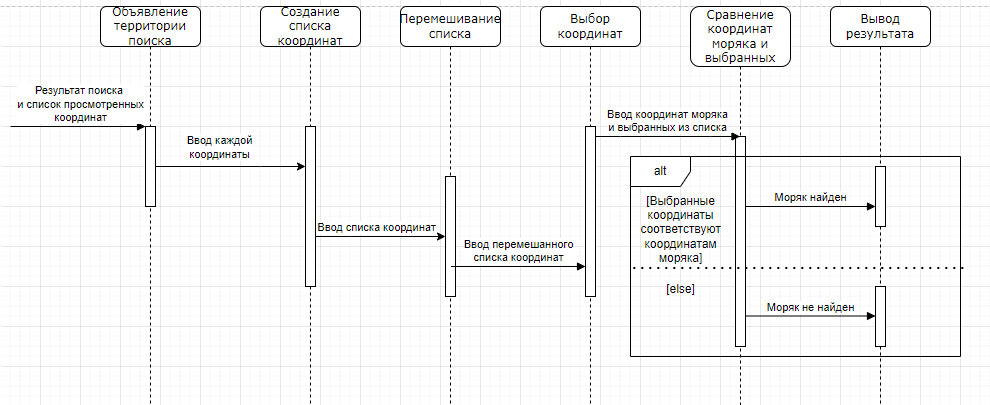


Рис. 4. Диаграмма последовательности

Код, отраженный в данной диаграмме находится в Приложении, листинг 1.5.

**БЛОК РАЗРАБОТКИ**

Для выполнения проекта в качестве основного языка программирования был выбран язык Python.

1. Он прост в освоении и требует лишь минимальное знание английского.
2. На нем нужно писать меньше кода, чем, например, на Java или C++.
3. Он достаточно универсален и используется почти для всех задач и на нем много инструментов.
4. Он совместим с необходимыми нам библиотеками OpenCV и Numру.

Код для данной игры был взят из предложенных материалов, мы изменили в нем ссылку на изображение, сделав из него байтовое изображение.

Диаграмму пакетов делать не нужно, так как у нас программа состоит из одного файла, в котором уже есть все необходимые библиотеки.

**ПРИЛОЖЕНИЕ**

Листинг кода

#Листинг 1.1. Импорт модулей и присваивание констант, используемых в программе bayes

import sys

import random

import itertools

import numpy as np

import cv2 as cv

import base64

b64\_file=b'iVBORw0KGgoAAAANSUhEUgAAAfQAAAF8CAAAAAAcXhbjAAAgAElEQVR4AezBB4BcZdn34d//fs6Z\nLSmkEUKRpnQERZT2Yi/YUJovFhRfULFhYyDZ3hOQsSIICgoqYMeODbBSpbfQCZ0kpGyyZeac5/5m\nElBKSDbJBshnrkvOKvzt1fC3vR9+ERv8f6IkZxV+8W448UPxshtkJiSxFlxURZcQo8OJGNFcblEi\nD4xcDgFwqsR/gzidkpxVuOR1wNX8rrk7leQm1oKLKo8CiVERHQ/E4O4WDWJg5CIS/1VikZKcVVj4\n2/fDXy/sa0kLAYuGWCsuyDEQoyIHTFHuHqLhHhi5CMZ/lVikJGcV/PYdWrsvfh3Ql0rRWEsuiBij\nxQHh5C5hcsTIOYgqd4z/CrFISc4qxCXfOo4r6neD5nGJuSPWQnQDB4zR4lJ0y/MsiUnibi5GzEFU\nRTD+K8QiJTmrEq/ai3/tMv/yg5saQwjRWHOOuzy6JxKIUeDm4DEMD3XR1pAqCjFiDqLKHeO/QixS\nkrNKf3oT925hj31r7CJTmqVRrCF3xz13pw4PiNHh0T02UdWbCiE2eFaxSEnOKt130YS4y7YLptJe\nb2mUizXgEMmJMQ73QccYgRgNLq9ksZ2anhTM2ODZxSIlOauSEW976dmH9P/42NbxHhyxBtwjOcOd\nLNc51l2IURGH8g64sPG+95/5KEYQGzyrWKQkZxXcuWMHOj7yvemcHIax4GL1eV6pkPdQc/pWDxzV\nVWdCrD237Hiqzn13MnD+NtdULAlixCISVQ7iv0IsUpKzKnH49M9xztu/2fy7G60sM4GDGDEX7oNt\n1PTtNH6XCem927bXB0OsnSggLu2ks/3nr5mggZ9P+XsPJVyO2GCFYpGSnFWJ8S9v5PYt5yyetE23\nJQSBgxgpxy3LmoGuV2w/tcFTHjqrtWmcErGWHBwf6ODq8S8q4PdtRdXXKtE8BjZYoVikJGcVIgxe\n+aKtEvfHNi5luQUDBzFSrixW2uAbB4yfEARUzjmaroLMWGsOWd7MH15dBw+c2gc0TfYYHLnYYAVi\nkZKcVXDk5UKUqJz/ofY0NWM1eT7QCRftOkUsU/nyCcw0ghgF7t7fzQ0728KvdtDa3RsS5QEXG6xQ\nLFKSswoePaH8F+01Vr8/gK6QBlZTvqgX/rR/KnCEx2v3ZOZwUiexdhx5FOVmvvO+woObQ1cqgmIg\nGhusUCxSkjMC2Z8P4I6tw/wffrKjkAQXI+SWCxZ30bzn68eDI/zh/sZrb2gCZqaAWHPu5jnGgj5O\n+kT6y8OgKzER3HCxwQrFIiU5I1A58+Nf/NBkGz7/SE4auxQhRsTlUJnBKYdMSkXNvde9q/39vyhS\ndZKwPLAW8uAeyZrh9m3/8Wpor8dMYoNnF4uU5IxAft1Nb54q4tU/7+Orw0JiJFwQ4wlw047mouZX\nB8I//9je5t2dDSaisRai+SDDPfDLt4WfHdLuDSaZiw2eXSxSkrMqLtdw1ijI8/tfTF8wjBFxRXk2\ng9JHxspFzfePoKY7WBKTXHKxFuKSTmqufpl+emhPTM1kubHBs4tFSnJWxZUHXzjcOBYPQ919nXUB\nY0QcV15uZc4Wysyo+f4RVDWNTVwycLHm3H1pJ9C5z/80+M8PaWpMzNyiscGzi0VKclbF5Vr8zRNO\n+vDkGIa6+zrqgszFCLiI2UB36/FjwQXEu/56VHNdTiGVExw5Yk25cmUzgNu2rIPrfhoKgSRibLAS\nsUhJzkg8Mg2+dPgmZN87uruAhVxi1RyywU6ufHlwkQe48aXQXpd7mljEHLFWouXZkj4enSKWnnVs\n6xiTOWKDlYhFSnJG4uFNe1q4encrn/XxroKZXIxAFIOtfPHTddQ4924D9JljCY5YSw6ezeCkT9ez\neOG5M7rTxI0NVi4WKckZiYc2K32BG3eyyrlHdqXBXGIEog+3wl3bAO5aumgLaK8PWG7GaHAnn863\nP5Twm/uPYWYwscEqxCIlOSPRf/an4dEp8lt2aWlIzc0Rq+IxLu7jrq0MiFpyzqegta4OgRglMZ7A\nt45M+O6HYWYiscEqxCIlOSMRr3853LNFYP6UlgZLMFbNY4xN/PCglJp439ZAV71QDIyWPE7njA8n\nzD6vk85GMDZYuVikJGdE7t4Wrn5pkucXvbWtPpHkYuWi8lhp49JXBhdwx3d7qZoly4OxltzIJUWP\nWTOnHZ1w1SuhsxGJDVYuFinJGZH+U6fD/En4Va9qGWOGuVi5XGWa4YFNFS1if34jNbMsRA+stVzU\nZDPglI8lD20GbY0BY4NViEVKckbC/d4rDu85dhxcsVdXSoLESnlkuLWjg4v+J5HnIVtw+YFAdyhE\nT3ITaysTkueL++BnB4a5UzstCcERG6xcLFKSMxLunl258YuD27Uv77YQ3MTKuCsf7Jg5g4enRqzM\nv/aD9oInJGARsXYcIspjuYPe3V41lQe2oKsuRBMbrEIsUpIzEp4lymW5acFXuroLkrFSTh7yE3pa\nmDtZ+eCPhj8B9IaYyCU31pYrDlqa5W1w/9QUv+sldBWMwAarEouU5IxMjA9OMwXPvnZcR33wwCpE\n5Sd0tfHoFNH/3YdmQnuDEYMcjLXk+GAby3znfQVcd2/7laWJB4kNViEWKckZCYcHXtS22XsmaP7p\nzV1JIQ9i5aJiuYXzDqqDOVt1ttOVBlmUcLFWHLkz0E7Nye/Z3KL8sVPrXZK52GDlYpGSnJHxO7aH\n21689KzP0heCm4uViz7QwW0vURw476MzK54GBUaBg3DPp0Pzm9Jdxhn+6NKLj2ZmMMQGqxKLlOSM\njN+xPVy3yV/fQ8uYBHPEymUs7WL+JBb9+YaOvmgpEqPBBeSUW/j73gRgwcl90DHGZY7YYOVikZKc\nkXB0+/bw98uOg96QuBCrkGlxT8fx9bppV+hJAsYoirnP4A9vMHAtOfPRvva61BzEBqsQi5TkjEy8\nfyu44qF3QfM4Je4SK5WLIl/9WB037wI9aeKI0ROjT+fn70iI2GOTT15QaPAkCrHBKsQiJTkj9Mg0\nuOFF/7rhs5xIcBcSLp7BLYLl7oMdzN5O9F/wwY46BYlRFGPexI/fnVDV/+2N7ktTklxiHXKL5i4X\n67VYpCRnhAa+3ML1u/iV+3CSIBKQixWJyoN7PoOfvLMAvuh3jy2MDR4YRTnHw8/fkeBw31bfWKRE\ncrFuRcPlYr0Wi5TkjEzkztkLDpiiK/biREMxJtFAPIMLKsKzFq7cw8C5ZRdaxigYo8XJ42BX24e2\nNuCxnx/dXYeBWKeiewC5WK/FIiU5I+FuebSHNhrH5Xszy8yLTePN3FihPJOxqI9LX2W428KJ0FFI\nAqPC5VW5Tb/nRQYsOv0EOhqDi3XLc4+JOUisz2KRkpwR8mhL0zou24eZFhZ30dYYzMUzOCzqpjN2\nUjpqvKiKs3eGzrqUURHNPXoeW+dsIWDBJOipw1jHPC5RIRGuwPosFinJGZnohoh26b7MIlnSQXu9\nJS6eznGK1DR/fDM5IM+vfSXNYxOJURBBeSy3f/Ho8UYe5m1MW0PqxjrmSzpoTxViCGI9FouU5IxM\nFNFAl+3DLMKSzqaEhgQpIvEkTuR4aq7dJYkS4LH8q/+lVAkCxHIu1oRDFvDFPVy3i+Ea+v5H6U4S\nYx1yOV5poqqpUBmbYC5qnCqxXolFSnJGyOUqJ3bZPswMx1PT1hAI7hJP5jFWWqn65W6TGk1AFPM3\n/ma+MJUbiGUcsQbcbZig4zn/XfW4U/7Ve+gzS6JYZ1xRWd7MzBnUzApuosYBsX6JRUpyRiq/eXb9\n/htdtg+9NoOOCXUfp6kxNXPEk7nDYFsvzfDFj4wX4JBf9vvGLE1c7mbUuFh97hCXJmls5oZdRB5y\nv38butOAse64clFpgl/Nn9NGS52FRKyvYpGSnBFx0V/q5L7N73pJ0xia6Tl83E8+SVNjISEinsTx\n6J5QhL5PjwXco8L8r6RZoRDyTCFhjcXcbbCjpb4F5k8EuTz7xWHNYxOMdcjdidNhztSBKw4446Md\nacFYX8UiJTkjEuWVP76T+6bd+lK6Ve6i5YNjv9NMy9hEuHiK6J4nDHTwt71SiEKuRzeBriSUc9Ul\nYk3FcoUOav70mgRXbsru2a51TBID61BE0U/gtA81MHzry2hqSBKMGgexXolFSnJGxBV5+BeNb7vz\nomY6vFDu5KZpk4HOOjPxFC4y4cNt9B68yQR5xMTQ7w6G5l6gt86NNeP5CSzX+8mNcPfg+RWXHkdf\nirHuODjZDP62T4Ch8/6PliRN3QycaGK9EouU5IyEI5b89Eh+c9PxVDVbN/dMOX9u42fa6yxIPE2U\nez7YRXvylt0KMVDVf/PeLPOlTEKsNgcvAi2v7T+o870vEeCu4TM+Q3uDmViHnDyUB7sv2TeNYsHt\ne0NLXZJkCh5yE+uVWKQkZyQcsXAiXHr9x3jcnVsvuOUvLR31ZmIFonM8VX/bx9wAH76i8kY4s9zw\nMGZitUU8n8EvH9161292/Phtje6GR1UuuDVvDDER61BUtIqmc9uLzZWXL7miA5obUCIhsV6JRUpy\nRsJBCyfCgwOXjRs3dsnrgbu2it/5KJ2FEFihPAxWYjdtn9xYLFO2G/+6Zd3DhbtTmbHa3Mn6++A7\n7/jlUdy3OchdPnjq8e11SUxdrDuuSIwzOPeQQrRoQw/d/hZqZoUYxPolFinJGQlHSy88lDMObxiq\nYyl/uGXaNvuOvXtb6E4siBWJYbhCpZv7NxM10cgX539678xyEhKx+tw13Az84pV/8rdsgiPX8J/f\nTlt94mkUT4iOiVHl4NkMZn5qLDWxfN3N/wd0pjE1ifVJLFKSMxIOS878HDw4Lgznp+2+LROnBHtk\nU1qVeqNYgSj33GMLv3xLgWWiMfir/wW6k1SsAVelXO7hmx+wJAgQ/sim0FRfH1w8IXr0EBhFbrmc\nbAZ/2xcRzY3K8H2PXbVAUXUpxvokFinJGZn4wJbArVeMmb3HAXDyURuJB7agprcOxDO5u7IZcOFr\n6gGXi/6vtp7U301vKszFExxR5S43nl1UnjXDKYdPkiOqfOj8/wNOtlwIlytLYhY9TaOISIwOV/Ss\nidnbyUFRgsi88z8D9BWw3FysJ2KRkpyRcDR3KnDNFfHjMzb7NDM/vtFjYfGW1PQWQGIF3IdbgIt2\nsCmJC6jc8NAeXPCJlnEhmvg3R1S5g/HsPKq/C0qVd+wQEOBuN+wGfHd+dEMQ5Zad0JqPq3OXO8Yo\ncfd8BrO3E1UuF5Df8peHemlrUIqL9UUsUpIzMrFy5UONUy6ZDvzrF108MvCtMckJQFdsNLlYAfds\nBsv94O3jRVVWqWfOWZsuQjLxBEfUuEs8O49eaeHCTV7OTTshwJVdc/stPafHfjcDlyM/Duiud5mD\nGCXuns/glh1ElctFVbZw6PJDuz2kwcX6IhYpyRmJ6MEreRIvey00e9/M/abuCN2twImeiBWLXm7h\nm/f1UvWDgxqocsSjm8x0CAnPEF3GSmTkTdyx2RWPvH4SBuQLrn8D0JdGM6qi5RpqBUpDhCAxatw9\nn8Hs7cQyLmqc619GU1KfBNYbsUhJzgi4o3uzrVIGbizv39JDV0MRaE8spijhWUTPZwCd7VTds6UA\nBz2wRWtCKJh4ujzKMHcwVsAjWRPnvMdiwZHw+8/oAbosNSliZEnFjoc/3/VAniQWAqPGPcYZ3PYS\n8VQPntYDnQ1JFOuJWKQkZwQcX1zquXlHDZ8z5v0scxLDnsrTXIYc8Uzu3t8NM7NIO7fsIKpcPLAF\n0FkfEE/hqmQKwXJ3BTniaXKL+dIe5k3G8+BG/1dboSMViSzPLSgPmU+H6299DzTV1wUxamKMM7hj\nW7n4N0flu6+5ZwYzEyTWD7FISc4IuIZvu6Dt2l1D9tu7PkdzL3TWEQlgYBFzxAp4fgIzQ+55K49O\nEeB5Epd86zg40SR3cBmKAlx+HB2hTjHzNKBoPE2WxHIzLBoPUdG8fO59WZIEYaZKnlgSY5a1cvoH\n5150JNBTCC5GhxPjdGZvpzywnKMoOUuveTXdBRNivRCLlOSMyIOb07pn/Ss5+3PQTgyyJAACBw/E\nwIr4cCyYe6WFm3cwqly5Pzz/xg+clCUWI3KJ4Ah3X9QL9MoGCnUhMxBP4SgOtPOH1yVRwiPzptFa\nSJFcqgw0pOYVb4YL38SSeVcfBjNTEKPC8bioj0enIGpcjlw88s/+D9HTmGGI9UIsUpIzAs68qVRd\ndNOngQ7lpiQRINwtSjEgVsCzPJU8n86XP9oIOCg3Lb7tV119iefuHhIJRMwrrZzyKTqyHvhyzJHx\nVI7y4+n71DhyA7f5p7XSkSRyHA11tNWLfKAPZm+HFjw854D2tCCJUeKx0kzbpyeLGpcz/FhSOOME\naBmT4hLrh1ikJGckKkv+evkrb/TXvJbHtdUZLrcQXcNJKsSKeI6B4qLe5uljqXFFLF5wCF115eEY\n8rqG4AiywQ64sbwHNV8bkIHxZBGPPoM7t1H0AGR37ASdFvOYxtBGVTO9wD/3qMuSPA7/aNLNiUyM\nFs8X9XHzjmK5OHzROzjng9DaoARjfRGLlOSMQHbpvw4a//v3nveqF8MP3s9yLal5mixVO5yYRIln\ncJTLPFre31P6RD0uwF2Kf3grrXUt1PQUhOHHAV/6SPLDI6G1e1aWBItg/Ifjldhy8sfGuhxFv+JP\nbfTENh73zWOo+dobt0siBotn7XxvYiZGiePlZq5+mVhu8DeHUdOWWmIS64tYpCRnFTwGr5z18X++\n6u+vvWjP3x7OxZv/atspf5/Bcq3dVPU2ZCaewbHM5LnHJr720YJcgLvEvPOOpap190NpGqdgXpkB\nX3/nljzyQGXju97cUigkig5JFDUucM+zlp6PT3Lh8vyMT/Jkt/RffQyc9dYpCY5g8Lwln5kZTIwS\nxytNXLO7WO6xydS0FwIBY70Ri5TkrIIj58q9znjPHw8750DdN/uVceumY9NrrilS09wLdKdmiGdy\ny0KufLgdfvaOFHeJZYZ+eCRw47T6PxzcqYKJcjO3bFMHjvPP/Wkem6qShySaqIkG5JRbeGQqVU75\nh4uOBb66w6SLT/jKZ/nHK5Lsjnv3SscEcEBDPzmCmWZCjArH8+lcs7tYbt7G0E5IgxBivRGLlOSs\nWuXOeftzw9AFr9uvLmbpoq/v+I7G4aGl/XPnvJ/uVrrSoJiwIk6e+uJOqh6cJlc0atx9zgP78/Wj\n6/3G3Wga64mVm/nbmB3rPQZx6/md9CZ5ntUVcIkn5PlwBw9No8q56yXU3LV5+MfFb5nkWxdEnhXk\nGUFukP3t9act9iDE6Igep3PtbqLGNW9jWgtSamK9EouU5KxCtHzBxnxn4r5j+jcqz93GyOc2NuTZ\nGPmd28GJS+pSZHKxIh4tPwHoeNfOBXfjcVGxfO28V2wq7//rO9vMCpbPAL72vkmCeP77v3lMU9IF\nPfVuLpZxeZaVu3hwU2ri7TtS84NDC0uXTAoLxtQhqm67bMGEt0805bf9oPeUoWggRkf0rIlrdxM1\nUfOm0pW6UrF+iUVKclbBcw2cbv87RUEsueNFk0KEax544M3b2CPT6KrLg0USVsgV8UozvVtucvFr\n99ooGi6qXEAlFkT0h7eAprGJLaIbrtnN8PiXN7DcyURJLOPE46l6aBpVzuDvDqXqD68pQJx76kvf\n1li+v+zh0iPhhl3EI9NoGUcQoyZ61sS1u4ka17yNm8aYkbCeiUVKclbBUWVO/TQpLw8v3rrloN3K\nFw7e0QEPbXLndnTWIcNNrJg72Qw6p30MrtzDHESVu+UBcM+T4fPu7qajLonlVrh8T7nx4ObUfOWz\ns2IwgcCduLgXmPmJ8SwTlyy8f9GWEyY2yJf+8WB4YNJvD2G563dV+eID6CikxuiJnjVx7W5iublT\nW8aYIcR6JRYpyVmF3Fh8XsOB41W5oP8j0P25eVtTc9XUOf/DyZbFYI54Fu4UWeakoydADDxZFhMt\nGpp93UKzpNwKV+4h4X95HVWnv/WRi4fqZMpN0d2zFjh9s5dtao7xJM7tOwDX1e9ATXsnt+ygxybT\nWgh1LkZNJJvB9bvK3aiat3FroxnGeiYWKclZIUc8zv2Rzbhra5aeutExwPXXfpB/6y4YGCvhqizu\no70T5myhaDyJC5ejO19Cc2KxE27eQdx52+2fBf74qvDPN7eMSVBuyhnwLrjwf9JEuPFUD53WzeO+\nOJh3Mns7btmlt1wIFhg9MWc6N+0klpu3cUtjMIz1TCxSkrMKrvLvDzzn4DEMnLGwE77+aaA1KfdS\nMzMxFysXszx37+TGnSwaTxalaHDLziz307fUW+Vbn6LmVzv/4vN0FFJ3mfuSLuDkozYSDqLGEcvN\n3pFlutr42rzQzs075mc/NHFenSkwemLuM7hpJ7HcvI2bG0PixnomFinJWYUhv35vrnq5sWAS8KMd\nd4NOSywbdq+zJGGVPMvdfKCXf76i4OLfopArYv1/vaaVqr/s2SDyy/ejvRN+PWVvmhrNTCIu7APO\neeM04WI5B7d5s5Ot6huX/PiYJvrgj4/e2k1THzftlH1pOvSlbmK0OJHjOf39jXLhYsGktiRN3VjP\nxCIlOc8qCrTke4WjOftd4+PAHw4FruHlfPeRPCjGLMjM3HCxUjHGNItN8NAmiKdwRK64ZM5lH+PE\nD00VztA9O1N1S/knRpKQorioB36z+9QUiBjLOJTjxW+Hr79/o8X/eiM1tz62L53t3LhzfsFh0Je6\nidETfaCDk46eIGqWnDFYrk8l1jOxyMlyVqQy8MjEezebZtlN2Z7Qd/jW8MCLqDpzwiFfmjw3M+Vg\njrmMVXBFN2dpBz96V4GniQYusod+ewy3bI8Q92wDXLj/v14Nrd19BcePh1+8oTK0ZMuUKGM5588P\nHEnVpZPColfQ4Z38+h3Q18R1L83P+yAd9UmeMIo8cjxcvqfhLu+fAL2pxcD6JRY5Sc6KnPEx4PT3\npRe9raVn1kETJ0nM3YSalh46CoWo6GbuhhyxUq4oPKu0wUObiCoXT5ctvKF//wnI3ZaedezJe+9Z\n97u3UdVbHz0OdcDf5z18zI9fO1kuqtwtxj+8nZqvfPbr8zvoja1UtTY0cdtL8gsOo7Pe3MQoij7U\nxg8PSnGUxyv3a5qSyU2sV2KRPjkrIqp+csDwZOCU900kN92zDV/dcz/aO6EvkLuZY+YWJVYquinG\nSlbp5a/7mSOeyUWZQKDKb529Xd02C6dQ882BsjIqHTT3AjfvCKImKj7yyN8+wzJNffQlfgLMGkzD\nDB6cxh3b053KjFHk5NlQd9cXGt2RGPrG4LiBYBjrlfx4euWsyD/ufT+nbL/VhIvv2vnugzZLHbel\nvxne4da5lbeWL/1MO54l9eSyICdh5TIJH2yn6vuHB1w8g8tFFMIR5cx8/hnd8LMpr+4xbNit0k3N\nnVtj1OSyRSf3XPqH9o4OoCUU0hArSwqJkqF2Fmyk8u8PPCknkRg9uWLMW7h/M0Uh8tt3oiNNCayE\nI15IXJXpzJSzAp5VZg9s8eMl9UeMGTrj5a8fgwuGuehtX/vAeHtwC2ravLupMQSXIVYmA6s0ATNn\nnPXBgItncEUM3M1RLmPBVzvhB++6byfa0xb+bf4klnMYOGvuwTd88PS5Mc9TKwSIlSxRbOGcw1N4\nZFpXSpAxenIR8yaaPzVVbkQNn39PZ1dqQTy7GM14AXEWdfNrOSviyuT/enho/7t/M/MrRzRGXb/1\nRvXc8+N99w4M/fiDQEsPVa2FEDy4ycWzKM9gVmyC7+885bJ9J/99s+1SosRTRIEc4fJoYunpXwDm\nbDF/Y2ju5XFfPGir4BjgDA9Vrn14pxvv2On+WLFgAfOYu4VKEz9/9biUxWerX0HG6ImyzPt7uOyV\nRhQMnT1nJp1pKjdX9ODiPxw5EPNg4oXCFYdb4K9yntVwqCw5/1h44Kab37hr5y5vr88GG4OLBfct\nkX7fwXc+DLSNH04IlhvP5Cj68dDUB7/bvv7O/CWXHdb56Qlyl1zgIFZk8Hsfg0v3KCyawHLnj2N8\n/TYTjeU8+8ubqPry5742kEdLTdGI0QPYF+CKXRc9tsvMmGDG6HHkeaWFC98YqMlvvf8AaBljCrm5\nC+M/HEVHniUmXhjcspANdvDd98hZMRcu8utn37vnnt8+rqXn5P/ZvWBUeTRBfsmV79z01FZqOuuC\nG2IF3LW4i6qfbLI/NPcCv5T2G2capF4uViw+fPdFe76mkfLlty35LPCPlxumIJ5Q/vYnqbrwgJYG\nTKlyA6Jys1iE0ovfDd1pcBOjymM+g6++f5KoKQ9d9hboVopCDjIXj4tyZR7zNvoKiBcEt8yyxX3c\nt4WcZ+duA5QLheuv/zC/fm2dB8NxJGBw8PYrP81yXQ1EBZ7Bqcqm09Ywo2egj3+7eyuW/HDpgVtZ\nTmCFvNI/ts7RcD73rC6+fsRGPFW87Yb3nPiKTbc49wHVm6XkRpVHIxvqoKalvmBujC73ONRO1xFb\nGssMPHRuW5PVhxRXFEgs4458OKoVmicgEC8E0cozOPNDQc6zcojB5WQDfzxsxrFTMWoiRtXt2/OE\nWaFipDyD4x5tKKtL+9UFLUm2UeMngUc3zq7eC+ZsbrmJFXKxXP9XW3++/0TDxZNlg3MnNYbK2ccA\nswoRBE5VXil3UdWZppYHRpEj8Liki5+9I6Ummt+xPdBRr4lj5w+AGY9zlrZT05uaGy8EbjlFuH5X\nyXl2jnB5Hnh0GtfvgrkAd8wZOucYnnDK4miW8nSOR49Le+kpxLwZuqXN4lHAtbvetDvwj73NxbNy\n4W7Ztbe+YZPcBMZ/RGzoL8P7Tpj9x88Bs9LoJhxcHm2g3EtbWgggRpEjcIabOfmYMdQ4Kv/z7600\nNWjyhLkDTggslyufzjInIYwXhBg5gVM/MA45q+JyVW5fut1GLnwoFHCLxtC5R1HT2g1fXNzdXUg9\nSi4XVY4ioEoc7IGmjfIMVwEfM3gCcOJb/86z/5IAACAASURBVP4J4HdvNhcr4G4uXLiblykIhzxh\nGUdArkWTuPiVC684GFoaE0fBWcYp54RAsGiMKheQ5wPdPDhNDigaw/fuAK3WOdNyD0qiwEVOpYUL\nxk98edNGyBAvAO4xa+Yv+ybIGZGcgMfQf9e9L9o1JRrlubfU3XY0NU19QHtDkMuRqHLIzeKiHmpO\nqlusYFmQx6yN5l6Wu+rl5ohnqFw6uNd4uVHjbkAUbizjCJdr8LLFr8h+9gWgOw0ZInGMqlhxLMgY\nZY7AybIWuj41QbmJmvI//tRHVVMfcGISDZd7OW/jwXH/uDRrTGTihSAqL7fyt1cVkDNS7iw58/Pw\n4DRBRLmG7yn88gs8rrtOcpCocpTL7Tjo3HyfXc5/dEGSEIlEWjtCK8uc9oExbjyTl3930FffPzlK\nLOfCQfxb9JCbVyj8+kCqZrn118mSgAuIuaIlrBseFRf18Y+9DLHc0GN3vprH9RViiPJIpQVuvvvt\nMCtIvCC4+/FwyT4F5Iycz90ETvlQ/dAYgQtyPXbesUDPLgfR2ejmuCTAkStXf9fXDphm3xteoFSW\nDedIPbR1scwd2yoi8Qz5I7/ddt86jwkr4iJWKo1ugh98gJq+JqC7MUpUOR7NWCdcecjLLXz/0Lpo\nLOf+wJaUvkBNZ6OjmKvSBl95z/mfp3lcygtFznArv31dPXJGyvHhS952yv/6dW/610vGaS6TDZh7\nw3aF9Oo5R9FVDyiXGTjOFJ9r2Qy+cUQ2Ceg2b+Epzn3Npu6BZ3Li0rQ+BlbMUeVP1x+2tQT/3A96\nph3d3Av0NGYSosbFuuGQU27lN29M5aLGs3Tu1K+86TL7MNBXiBaXWAdwypTN96e1Lk14oYgxb+Lc\ndzcgZ4Qcchu8d2r65U744pH5NG57sYFXEos/ei/MBERGGnA3Z0r+mPx4+NlrLj6U5VrqI+5bLPks\nLT00f3Ia7oEVcEAuViza0Klf+MNrE+GLLlr04Gv2p+rk45iFSWLdieC4583w07cXFI0qd3LuHX/3\n3m1ddKVJyLSkk6oztn7zd+7s6U6DxAuAK3rIFvVx7W5CzmqJ1v+lDmDmFkdwx7ZiGb/n2oM58+FM\nIcviuIDLcq/PY6TcAqcdkF31XqDpwxOpqc/uv3Wf+/dqOWI7EY3VFcuLJ1xz42Nv2jW4MWA+/JuB\nj8KZR9FVZxLrTiRGkTHUA7dvnRCNJzhX79na3VlHqnyolaY+fqJDaA5pIbjx/HOLUXmoNNFx7ESQ\ns5qy+x655hNA78HbFqhxkWfX7dXRwTJt9UmWmOMkmce8Ba54eX7XVfnit21rLJdnhezWsMU4EY3V\nVbnwwHPnfoaTPzZGvvCeF4/zu/3eKyqTP9U8NsFYh6LKmfIOqq7fVa4s4T/mn3lCU0OBJPpw6xdf\ne81HqWquK6QeTTz/IjGHJX2cd2gCclaTR/p/9UHo/nhDIaHGQf3f/jw1zb3N9WE4qQsxTxQ9z1vg\nupcqLpWPMf6jooDAxepaOBF6m2HBBPj9AT9+56JTp34SaB4XJNapfGknNb3v2ClB5IF/88r3j6Kr\nECyP/b0Xvu6q/YDWJE0F4gUgera0F/jWuzYG5Kwmj6bHrjyAc/NHD9mKKpej/N5HFzYsPpCLbv0E\nVV2NnieWL6hvA27ewVjGETUeLZpc7hKrq/ybg89/6HOce3BqXLPHr19+dhNVbWkhgFhnnDjURlX7\n7q+ZKNyNJ4s37UZHfSDS3/PPl51zDHSERKnLEc+/vDLYQ9U/9grRkLOaPJq8PO/RPeC6Lccbj6vI\niY8sanj4j11AZ30Soy3upuYbRzaCCxxEjcvlyN1YTa6hgf4pd7xs1gemXDaGSYPpHz9Fa3dX8CSY\nI9YVd2LWTPe26X4bJ/LoARBPcC//+IPNY4Ni1sy1ya5dQ3WJEhnRTTyvHNzLLcCsxW9+xVgXyFlt\neXDR/82F+4UDbtjJRI0j4P5vd1LTNC5xy2ywHbh46XYvDpCbcBSNaKwZlzPvvmv32DWp3LzVuL+9\nDpgxE9qVKKCgaKwrjkfPm3+3w6YqiGWi8STR7tiO7jri0m7ufWjv3iykMtwc8TxyrGIx92bg+il1\nE4Uj5KyZ+HAy/heH//wdgaF77nnVOFIBt+5EVY8FGTiLeoCO7QuqFF43NgEX7uSJS6yBKC09rchP\n31GIOX99I8u1pklQDBLrknv0wU7u21yskGvpdxdYGmMTPHjfXu0FSySeb+6EzCs+2Me3D5hSx+Pk\nrLl7rn7Vpj5w7sfhgjc1UnX79i0TFjWYPJgq0mAb7Z0sc8NOAVwO5A/O3X4cuFhtnl3y5m8ePCWb\nM37sOcewXMuYJA8eWNeiD7Vx485ihVzZT7nb4tIeuo4e2rYnWBAunmfuxKF2qm7bDhba8GQD5KwZ\ndyOvpDb/zOlUXb6nATfvQs+YQZNCbuD5kjHW30PNv15mPO6xyaccPplorD7X4MMbj8l+/r8X73fn\nTizXNDZ1DLGORR9q49btxYo4VE4tLNJQJ1zwhjm7tDYmmCOeb16ZQc3P3lY3uPCcSR/99SsnpJKz\nhqKEy2/bkZqrdzdg8dnH0pkkwWNwx91leeZ5pY8r9gg8buGXX/XmlGisPscFD23O797ovzwU+Omk\n19Gn4GasazGfTttnJvEs8uzkli7aKL16t+y6fZmZeJIbzy+HuKQLfprsObnQ/8PHpgM9u75+nJw1\n5XJc/dc/ULh7zMRNXjItgD/y0B501KURi7mie0hi7iztY/Z28pyIBWX3TpqgiMRqikJU9Z86/OHN\nmXf/w2MbrvkYzDRDiHXLY9bET96VsEIO2Smf7+iAezb3Xx4KM9M8zQK4wMXzw/F8SQ8/eOuY+cmc\nSxe2sszJr5azdjzPNGCP7NB5xCb1iHu2paW+DsjjcMUYk0aiz+BbR9QNLpy96L65Wx88zmWsBXce\nyjYLYvEllQc+Cz2pSaxbbu7xeLhmd4GjaDyNk//6IODU/6sbKrXQVWeYg7tBlBDPBydWmmFO4dRN\nPglNfR3y2EWXnNHw2GQ466Dx2Pwp0FFILFvcR1XLRCeLTVy1W7zw3dTcsZUngFhTLsctirtfTFVX\nwQLrnvuSTr7x3vEyyAzjaaIq3/0Y8Iu3JbfsTGtjgrnIkLkTjcDzw/P+HpZrtxBUqXTRKWc0zNkK\n+PKRGyn7yXuhvaDhTqCpr3mczBf28a0GP4JlbtrRCVFiLbhyLbhx3qFAeyEJYt2L8QT46da3Tnhp\nYXw9MfBUWVB26+VHt2154JThS19Pd5ognDyTJZl7ai6eH7Ey0AO0JKSyoKzSSp+cteWI+VOoOvUD\njeHh2a+FNlX6qPr6p2cmOQNdLNPUB9zxohSxFiLm8qVnfQZaErMkGOuex6yJr3/6jI8C3z2s3nia\nqDyUK/Nsal3l9wfSND4YuDwbDsEW9fWGlOdJJPpgJ831CQGEx+mcJGcUeH79fbdOh5u3Szy7dTeW\nueIGjqKrHovH84T2Ti7ZzyTWUrxxd/h6NhyDTKxjLjyWW3jCLTuIp4lkKQiGr94XOuuDIMqHWmlS\nLy3ThgDxHHNBVPQ8VtJEGHKIJ1CSs9bcjczL9+/Y+aEtXcpv25mqkz519Zz30VGfOHEQr9AHLT3w\n51cnrIVouBg+Y34exkdcxrrnlqlIVYdT6ePrHx4Drmg8hbtRvvBd0NpoCXiMMZxAzQ/H3RwlxHPL\nBa6omLuZPIBDPJ6SnNGS31eYGhDkN/9529uP++1bfvRe6KxDFitxuJeqmeM/yd1bucQa8WgRyzK4\n7PUd9WVLDYnngMelnUBTgywb6mPOFvIYovEMS08Z7GwvFMA8eqUc2n92MNA2IUPG88VzFBxRE4uU\n5IwaB5ccZUsaBm7dcaNLrq17uKfHAllsZZmWntP22aHexRpxRNUNl/SH6TArSyzwnPCYl9uBjiT1\nmLdy+Z4GOOKpnKGzF03vsVTuFivlTvjLa6jqCwHxvHEQj4tFSnJGk4uIUVVJ9OCpvUB7XWyh6lsf\noeorxxTkYs1kt9VvkWQ/PxzossSDjOeG5z6dnx5CbyLyGfDgtGiAeCrXgkm01YcEIpWsDTo6qGpv\nNOMFIhYpyRk9LiBmS8Yn0YgaOOtYqk5a1AvcstntewLfeV/BxZqpXLUvf9z70jcDbSEN8oTnhpNz\nAlUzZZaV2+DBacoDT+dLf/x/dCdJNGXlNuDMo3oLg+YhxXiBiEVKckaPC4hX7v37V6cyJ973vTbg\nu0fyddtp/+S2HYBLXwXGmllaaudn+28MtFooWCQYzwVXdA20Q1djHhR9US8XvD1xEE8V52wDsyRX\nrAz3wBe3PqytsZJ6gsSTuCPx/IhFSnJG28Wv51+7JQ7kcxdVsl92wHXb1gcW/+CTnHboZJcbq881\n+LtD4F/bn/lZqtob8qyQCDlinXPIBzrbGhPAfaiN7x9Szwrcvj2ddRY9pwX40fU90Buk4OJJ3KNb\nUG48D2KRkpzRNve8z/x533pw8gSyq/aB2VvOnzI/DbM3njJeyMVqc7H0S23wcDqZJ7SOTSJCPBfy\nSh6SIEf5cCvcvZUc8TS3b0+vqUIrNZfvBa2NCRJPFlHmlhCN50EsUpIz2vyxBRuPl8vliJjdq7pN\n/vH683Yev0ma4MijiTWQXfvHV+wyrfLXt/CEmWYI8VzwCrKAewyVgW4ueq0bT7f4yx10JsNd1HRP\n/Ri014Ugniq6DwdL5eJ5EIuU5KwDDywI46eZS46ouWyfb08rzDlsnFwuF6svStlwfcCHH738pQ++\ngZqekDiB50YmTETHcp/OOYenjniquVOhs9wLXwwDLdTMDJ64eIoYKy0wKwHx3ItFSnJGX+UHH+Zr\n790oiRLkicPiu644Bk786ATcjTURJRdEi2RX7A8tPfDFKGQ8N6IQROG5lnT3fb6OZ1g4sS32AH8f\nt/vZd3ZBR33qLvFUeaUZ6ClIFsVzLRYpyRl9+e07QfsuBzSaIwR5vOTNVD24KdFwsSYcgePZoivf\nTmfqrtRciOeII3BFxVhua2qt5xmWfutzVN04cXP6crkSMxfiCY65ezwBmvpmBQ+4eK7FIiU560B+\n48vg7HePEzWOD512HHD2YQ2stXjJG4DONMkJgeeBkw+3ceNOxtNV/v564P+1Bx8AcpYF/se/v+d9\nZ3Y3m0I2jYQqVaRIUw4VFVGxHSIigqeed96dXU88SraX7AZEVP7CURTBBh6KKIcUBT1BREGUIiBS\nhCAhhfRNdndm3uf3n90UQgjBJJtkYebzeajhgiZaahKRyEKATQADclZshtbakDpgsbXFUzhbZthF\nlD1wAI/slCLKjEp/Oojz3zK1DrHZvvlRYGYui0EJ24Jdmk73J8eJdRRX/PpdPR+cevtraUprQhIT\nGQfWMJD1Zj3w3d5FKRJbXzyFs2W2gFISH4q712QKDMqkwsMrdp2oSGDzGL7+MRpra3IlxRDYJhxX\ntHPngYF1OF593Lknzv9hwaFWKTEFI9aIIVNvBzT2XF+8LxAQW108hbNltoAoZYQsZSULMieyJTaP\n4aKP05GmSZbGwLbhrNQEy0exjkx/veOAve85uLuY5AlWYA2LQTGeBmcc/NZfDtylIMRWF0/hbJkt\nIErGCNG7dPsAGOEYHNhs5326cbSSkKUW20iJgVaensA6okvFUeEPhzTV5JMYQwisFk0CVnYqfP/N\no/7niIf/qERi64uncLbMlmJBXHjJqde+sVazbz14VwlZbK4VZ7W31TsEi23FZqCZuZNZhxH4roOZ\nkSNmIS/AckwcSyGEGChO5/wP1c8759in7yGR2PriKZwtsyUtufQ/4Ylp4bp3nPuROiE236xdmFGD\nxLZjXGhi7mTWYQFxwWRmUmhjZpJQFgGXYpKGaBeb+fVrFl79r+eHxSSEKLa2eApny2xJ86YA30lf\nkf/Zaw5IZQKb7aG9mkelAbHtGBeamDuZdVhAzK7P3znQCU2jcgFkFEvT4YzgUmyBJ6fNuqhmoD4l\nsSy2tngKZ8tsSYWfv4tBd+w2Jmc7iM320F6ttakkth0Ti03Mm8Q6LMoGLhr3cBc0z6AnZ6EYsqXd\n0FQXi11w8Ymjnrps/OM1ecli64uncLbMllTqu+XuRsoe2BsHhkHvhQP5UhIQ246JxSbmTWIdFmUr\nzuyk7CufZ2aQkOOKDta4Zz89ssels0I+MWLri6dwlsyW5Diw9IefAa47iiRgsbkWTLx0XkYaAItt\nw8RiE3Mnsz4u/s+HYeaKLrryChDhVM4f9c+Ude1++M5h4MFX3vH7RWkQW5ll8CmcIbOFee4jf37K\nb98pHauEmLCZnp7U5ZCkwcJi2zCx2MScKayPeerxJ94PLblcKkE0p3Lj3nNuOZlz3z05DYXr383l\ni5cREFuXschOY4bMlmQZYmH58h+OHXfob/ffs5bNtfSMmY35fJqwlRixLhOLTcyZwnpFxXnTOtqY\nkUsEjopLei49Ide73GPGKuu9+RjoyZtgsVVZWcDFRtpltjALGLiqNPC6l19w0lg217KzO6B1TACH\nKLYsm6iEdZlYbGLOFNYrEuZd9nnoThNBJIYVbTw5jUF+6LIOaKzLpxZbm7M0Cwt76JHZ4iyyRaG0\neO+z/n0cm6v0+zs/DWckyEkUW1aMJiSsy8RiE3OmsD5ZYNk3vtDW0VwfEkyUi8185d9GUzb7zhUn\nQofySsRW5xiy5Z1wl8wWZhkLWPqH3aalYvPYxYF7fnNad1DqwJaWZVYaWJeJxSbmTGF9orhvf3qy\nXJAcs6CsEa59a0IMS77WArTlQ0hjYKtzFk6Bxg/uI7O1xGIueEU+kdg8fRd/pj1NQ3BgCzLy0k7o\nrFFisTYTi008tT3rFbVgEi21SSJif5bY7fD4TnJhwaVNQFs+lYLFxorBKKIYHNgENvE0uOPAVGZr\nWnLZjkfWKbAZrNk7QHtNGmIAIzaexQuyQzyFsu5ccGBtJhabeGp71s/x6uPoShJY2s2QCz5YnxVu\nWP5BmnMhSSWx8UoBZTKRlMCmiCr1dfDNE+tktqbZO/DYzmLzLDunBbpqJBFNEBstk4NYh8UzLENx\nOmVnBIIY5CgZgonFJp7anufhedf9S1tOITax0sMvE0vP2vljjTVpSBTYaJaLiBiIhBDEJjAq6jSY\ntaPM1tR3Y90/1CmwWfzED/6LttqQmJgpRSA2hktRScLaLMcgVnMUOC7tgeaxBESZY4xCShRdbOKp\n7VkfCxY1sMqVT/znV46aOj7B874f5uZySUJgI5ghlovFXFawYj7WJ2ySGOxiI+cfIbM1xf68AgIs\ni01istk//MIZJaWOJYe8REwQf7foYnQ+x1pslUgCK9kuSZAt7+l2Ls0Si7JYKhGVJiE4KzYzeyrr\nZ/qu/DArXX3oY/uME85mXzCTltqcHMTfxZZMtBFl2UBHUzdDOuoBsbFMiKa/hXNktqoYwKKwYmyw\nwGLjGc++5hO05/s7ga6a6JgqAYsXYhkVTwe66oRF5iSGLInRkVzAMnKklCFKHdCR5lASRVmpONAF\njdsFHIvNPDmN9bHAC2+e+wlgxgFH1scEmPeLBZ+GznxIbfECLDDRQRlZBILJWhnSVTvxX77aHySx\nsSysWGykR2brM7868jfbT601ltgUpVveRGsngzpqSirlEyWOwWJDHEppSSzrBDprgnAmh1ISA1nW\n0plXlCxUcAvQ2ENZez4EBdny0k4Gna1iyJb2MGsnnpf7Cv1FcvX5vIDCivtfC821aRLAYsOMHDI7\nohhpZlBTN1zxaDzgXZR15aXAJipMp1tm67OvOr6n8advyUUnbBIveeA1lJ3PJ87+AtBeT3CWWmyQ\nHbKktKSHnzw4dmEguBTBmeVSDzQrSiih0M1KXS20pCFNLKTFM+CSfRe8/eIlBfd3wNzJPC+Ltfz6\npnagM80F/h4xKYViLGZpMfawxiVvXzrnlibKZuZAbKLSacyU2eoMT9w2+l0/fNX2qcQmuvtAvjht\nh1ffdExjD9DYYIcYxIZlIQaKy3po7exIkpjEYuxgfZr23o6p89KjKetIEmISsum0vHvP3MMH9ASy\nJrjl1XmenxFlBrH0Z++Dxnwul/D3sENGsb+LIY27jOrL56bMr390cdYFbVZSm1hsEuN4GmfKbGXG\nj87a41s7abtjv/fuejaNWfC93ffp36Pml28Czsyd3CORIGIQz8uomGZxOmUtThK3sVrz+CkT3s6Q\n1lzL5W8YH0hL/T/8KEOa0iQXm7jlsOSad9OaZ0UP3zhhDBYbYEG0759zWzu0pGmqIF6AiQFFF1b0\n0JU7nfMO3XlcWPJoQ+HOj1DWWC/FJAkWm8A42tPpkdnKTPGSj99wr8a/7MiLThztwKYwXlbX3zcx\nmf/dk886BehKEocYVEok8XxMVLH25LP+ofgmVvl+Pk6eGAqTa3LMiUFykiS1dTWizEv7Hkt/UtNC\nWXvSwtVvDzfc1diaDPTAX3ZHiOdjgUX0XYdS1pUoBCReQIxOwMUm4M6JC6bW1wfmXfGZL58MNJKm\nOaWZFNh4BlPKcAs9MltdvOVnx28X0gfe8rWPjDZic3ju4prdgK5EBkIx5CRkxHMZm+LYefu+K7vi\n3xl07Q67pyQpz8OU9KRm3f8xhvR8uv6Kk2iiG377ylojNiyK4vf/GejMhwBiA0yImGKWkz0dbpqy\nWx1D/rwPZa2JRJKEYBAbwyLayHaxSBdcKLPVua+3Qej2wy89vg4hIzaNZS986gBoTUn6ZtDYQ+uo\nEJAR62FDMZzOg7stXBjnLxk3aY80sF4GMcTE/r7Fl3YzfSa3HXLhZxh0wxvzRIkX4P4bj6E5JLmQ\n4wU4JhGyZQ61YaCD770vCaz02OWN0JpPsqDUQWwUgw3RdqS/m0F3ymx9Js6enFv66E4TYsIgscmW\nfK0FmpRmXazUXpeYYLEexl7RDucePm6XtJQlIbABRqySzfp+Y2sndxz49U8CPdlHpwSMeAHZ4onQ\nniPnlBeSheg4nVUe3k2UxbkLir2vB85bkTlxiCkbJxpiJrK+Hga1Tarb70CZbcBLz95+xzfVERCb\nadFZM1mla8qyLwBnJHbOFs9h2fS1Mui7x4xhgwxijTi/b85fDtw7fWRvuHXvrCEFixfiPx5CY30u\n4MAQg03Cuiw7K7kFzv8EZde8uYZBfd/4LGXf+dBVjxeyQHDC388QoYRXOOli0Hmvm1afpEFm6zP9\n38uVjh0fjNhMhYcefC8ws7T/nlNqB3oLe3bmJRKL9XKMpzPk6n3FKum0wLrEuuJAnlB8pDaZlgJW\nMccLKV78CbpqgrKEIY4WMSSswzEoW9He2AOz/rJooLDnQaMsAb1jGPSj487pi8XSKCkE/j4Gm8zE\n2Mqgcyfv5P1HM0hmWygtuvEDP3pnns0XSyuWLMrXjInj8wKKN99ZyAvFRBbrioljKDnLSp2s7Zrc\nkowhRga060EKKyjFqBicjE1lAVElBQKYmJRSXsg9r2zM5WpiMAkQTUkxdZJEWaxhxVKigWbK7lx6\nZGMPF/xLnjLriZ2hqZuynkbKOmoTYbFhNpYd7djCoMaD+l83MZ86YYjMNhBDdusbbjoixzOixDDo\nv+oD9OQcJRRYl4UVowrEVjbsnnk1Nw90stJ1R+XYFLe+riemaSlLg0SMkQwVldYgI1ZyKCVFm0bK\nrjrquvcD1701MORnR9Mait3QXns6g85IHEA8LztaDiU5FjpZ6cwPTCnWiTVktj7HhKVzpo6KpSRJ\nWClKDIPec6e31qaOBCUxWDxX5hCtrJRZwsaKM3iOlq6vfJ41rn5zbUzYCBZlfzikaXTiYrEmlR1K\nhTaGdOcdxCoOxXAKzGjmweTJvSfO/WXN3F2OGIsYdOXxzAix1N5UlyvGrIOuGkiixPoYiDjizCW5\n1EPZxbuPz283KWFtMtuAHXDp4d/PnbDjXjUxkkwhMAxiyG47orE+lAbqQg4Hi+eKDoZYxBJYDqVi\nKkuAKbOy2AW0F7th5vIZlN1xiNh4D768cWzMWjjTBgptrNSTwwmrGPd2QEfbBR/OPzS5wYtzMV9j\niUFXHt84VnhFmg8iLkn3WrEEHMRzmSgcYwxZHOhhyLcmHTI2F0A8i8zWlyUW3HsAa/zhlYFhYPHE\nzrTUNMNZmMRiPSwglpBisMAxSjGwRgxugqZ8IiuJuL+L694a2Fh+4s7jmFHoZEiLZ0BHW0ehTvmY\nZgmrRHwqQ56afHvfq1OnKdGJBTw9qSWtJWRZSCFzqWZy3/IoJJ7FshUd5UipjZW+qIaXH5Sm4rlk\ntj4ji3n/3cFq17wtsRgOfV//HIMu7C2GIJ6XIwIssCKDhMUgB5dKVqrEUYqh0MJF79yuTgxaMKpW\n/F0GLv4UjWknfPlkhpy1593j08VKgoVYycRiE0Nu2+m7p59fu/yAg+tUSgJlC76x0yxJZKmVxehi\nJ2cESTyLDTHaRQo9LV2Ute37iobRMcnlxSCbwFpktgHLIs6+81gGNXXzv29PGB6+b38GzSTLJWID\nLMsCbGQxREYGssxKFBwlyIotcO4b9s6B/7bzN/Y9JEeZxYb1dXfT1gGXHPXo/Qta4Jtv09S2Wjkk\nsihzMIbidKCli9W+ecwEYqBs7nmveAxJBhcoqQN6UhEACyNjuSw6y9zJoC8enteODXnx/GS2EcP8\n73+Oss5Wfnp0wjDpv+fhOeN3+m0LTaPyEpvGNkiykUzW1w5c/rZx2fwLO+B7x46KBIsNK3yrr+90\nuH+PtJjRG+LY7Hv/QXstSlklKkS72ASNdaUOVpu1o4hOop7Y5dwlVpAc+zuBxh44KyoIMIKYBTKy\nONBNWVtx9D51hzSYIDZEZlux/Lc73gt0tPHToxOGScz6e+t1zQeAM0OIBDaJQZRZWJn7it1w8QcW\n/e/HKPvbNFkx8PwsKHxnxWfh5sNTBplZN38YZuSkwColkkg2nWfr+Mx4MIp6dI+WHKKlvZ3VWsYI\nCSxDJJI1Maixh+v3zNWPdY4XJLPNVZg97gAAGAtJREFUlJJYWPT0K+lq4adHJwwTxwTivPtKb6O9\nNolK2FxWdLGJsqf7f/RZyv7ffofVhkjgeVlx4Oa3AdceMZosK2Y8+acTgK5cgsQQU8oSl1rg3GXT\nWaPrQztb9MdR6j3/VKC1k0HfegPXqTR7dC5kQRhHK8aBrJtBlx4yKpmWYxUjNkBmmymmiIFzTmue\nwbVvTRgeBgfHJFt26ecbRycOQWwu46VdQNPrftdOa2gH/rIHYgOsuOLrJ8PP3pTAnx99dH4nZa2h\nJmeJMkOMp7fkYxtcf/jiP8wpNiysnfyH3K5v3K6euXUL/zj/sD3CFf/CKl1vOyD/9E2jpv6qpDQK\nG2emVOim7P/t/OqGvFjFKEpsgMy2kjmJCdz8BuD6NycMjygHwMquPo4OhZCThDHIiE1gIqdS1vWq\nhz9NR6ImeGBvGfG8jLI/HQhP7CAK1xYfagIaa5I0NQFs7BibGPKt99fEgcwh5jUvGV8j5k8+882P\nH8cdBz69aOlf+va893PA/IZH9oKvFrIQwNGOxQ4GNb3szZOTnBhkYQQWGyKzjd15KHDFP9Yghtes\nXWjrgJlJsDAoZBKbwu5vYaVG5fNMp/WzE3guC4uVLAq//9vu+9VQ+MXbufSh7qY8IU1QcIgRR8f+\nGQz6wRsmsZoRsGBi56uWH3/O+7YHViwtXvMp4H9HHwl0J4gsxFIrzTMY9Pvd8qPExpHZxhY9fm0T\nnH/SGIlhVbhx1ieAjpoEkaEEIzaBHU+j8VXvYaUut/LrwwMWGxazLMlRuv4f4aw+EuScHYRxKWtl\n0I9epvxuNcTAWmJxlsf+9N8uf8fo4p/6S3/+OKs1Up/EmOF2hlzw2lJ+t1o2msy2FeUrTwCuOiYw\nvDz3gg7KuvJSzJykMYkSG83EbDrPaEo7WNAApZS/x5M7AjNSAgaXkiRS7Es7GXTVUXWBgJEFMdhS\nlIP7eh965DVj59/yCdbWQprEVob0HFQYfegYEyiz2Bgy21SJlHt/2AlfPWmSGE5W4Q9/fSRphM5c\nKJZCzkmwJDZahFNY5dKxx7V2Qm89WLyQgfkDvUve0EwulwYTHXt72lNn7ZR9ZZ+dGsbnxUoWWEZE\nQpZd/T42oNsH77tdfVF5MciIjSKzjUUNLLz5JOj+XD2WxfBwTIrKlmX3vBXaQ38PM2NCIIiNZtSX\n9hVr2uCmA/tu+mc4+6S60QmI9estmiGP3RCboccKIcliFltY5YsTJr12TCJRZhlhS0AWtLxw/9gD\nWK3J6mZtXzpoh+09OmUlY8mIjSEzAizZDrh/ryRLLIaJZQF91x/Hat05EkVZbBwrczHSys9f+7sj\nLz+JsguPneTAuhwTzLLvfoq1dORDlqBiE2ucd9zY4sIpdWKQHSzAAiJ9P/wIZ51CU67QQyOom9W+\nOLlhUo49RweexWIjyYwAfaMou/2glGKO4eXe2+/6L1qOLf7lw9A4wYoSG81ZiSaYN+HH7/36vCag\n7QtjjFiXye57aOlHWaNNMac0S1xqoeyrr/9Jx6UHbNcwVn/eZ+bxuwUMCAssYPFDvfd9hkFN7mG1\ntlEHLCvtP3pcbU6QsNlkRoDCT06g7L6XB4aXLQ0sumf8rg3L5j95JC3TlmfBCRuvFJnOt09KZj+w\ne/5374Wuj00E8WxGrDj3NGgyMsj5ELLUyvp7KJs+8zcHPzX7gHoBC6/4xO2HBMeElXof2mn5/DE3\nfYpBjfSwUktXzwEH1ufyRdcFhovMCOBFC/ai7P6Xi+FkKOTlYk6LvvM54MyaohQlNpbteDrXvo3H\nfzV/BxV2Kk3aNxXPYRV+fRTQKWGQY4jR7mLQF0+oKUyujVkqIMYFS8end4+ZsEvsY9CPP/jFU3m2\n5oOnjRlbqtsuL4aXzIhQ+uk9rXDxB2qNGDaR/msPmNAglv/xCKC7hjKx8Yzjafz0aF32IcpuOTwQ\nA+LZrMKN74SWXCphHJWphSEXvCK313bBIgYLsMPffnAyZ5zY8MTPBoyzFp7t0nGTXz4uIItnOAax\n+WRGAFsLlu4OPDlVMTCMFk74UvrBCQPf/wi01IZEAsRGM85O55q3/GV/Bl3/VmEQz+bS/74XGkcp\nCUTsTP0zgNZOvn70NKcMskMMgLXwnE6+us/9n2ddTdvroF3Hp+RYLUpGgMUwkBkZ7Af2g++9t8Zi\n2MSw4m7rwNoVo6GxNi+CxCZx9Glc8cbJDLnxTeJZLCMXb3oHdCaJpUwx6++h7IzjJ9zzhu+9fkew\nA8bBslgWen/xgbYOnq2xp2v/g8ak2ZggthyZkWJRQ2snf90FMWyy4JJjjVac2dVYlyYQxCYx0afB\neZ+C1qyb371KPFfxhn+EjnzAWYxuZ9Al++w4NSx5bOq4PINkZGJSmPPjcR+B7iaecelOuWT7pVO3\ny4stTGakKN5852lc8Y56i+FiKD7020OnjrnqgVwuBxKbyPaplDWmncC9+wosnu3hPWkJuVRZ1sKQ\n8w8s7T0+EY7zb9pvjzqwQlTRT9fc8MR01tK1//Ipe0/IQ4hBbHEyI8bCCcAf90sZNlni0uX/3NL1\nf78sjkqSYAKbyPapQHPaTtmf9xJYrGHQkm9/lpZcUNYKtHVcsvOeDbUJZTEwewd+/ppRBrn0f+nP\nZ/KM6RP33HnnMcWQF1uLzEhR4scPqOXy4/IMnxjDg79f9Ln2dnpCIhCbyLi3AzraoHv5XseOs7BY\nw4rZ5Q930eqYdFL2QN2UJBVlpqxw98OH7ZIIBuLNb/vyyazUOKah/uWTRo9OElayMLLYsmRGDD9x\nZX8jD+wtho1xqXTZv3/tM81jQrDYNEZgF0ulTug5oX7MKFmsLcrZDXd00thD2U9r9twhYZWSJFQs\njLKD3Hvb0azyjf3HNozN8jmeYWHEliYzcvRecAp85d/rGV4X/1v3QG1OYpMYMSjaA8UuuPPAwHNk\nIcYfn8CgL9Xu8fokDZQZYbGKKZRG8cAP2oEzDp2anzJabBsyI0UpzR7cF5i9vRhWF/8bHfkQJJ7F\nYCQ2zNGJcHCM9LXDD96TsB7R4ZFf/seXT+bnhymtEWtYrGQGbvrbUbvGefNmh7oDxguxrciMFIbS\nvbd+lr9NE8Pqf06kIyRKA2vYIjoSQmDDYlE5OYZYilkbfPd1u2BnOdZlin1PjVscd6kXWKyWFR8J\n8x2X77ZXuuxrTVceOS5QhFRsQzIjRnQo9v4hfXWdGFa/fBO0hySXsJpjJGSZnUsSNsTOSkkasoxS\nS/MM+OmbczI48GyWHWwZibUtffR3H2+eQdld+8b75++zvYJYyYoEtgWZkcIyaEB5htmP30NZ09gE\nxCDHUhZKbcCMXMoaFoMsVrKis9jUk6OQxXbKrnpLPWQhKvBcRqzFpUf68A2NrHLV0TVZVmOcUGZk\nxLYhM5KYrHdUTgyrR2//Sxsw02nAgWgKK1A3ZV8CSwxyyBxwUCZR5lAiurebs7S8jbJL6/d/WU48\nl8Wz9RYNC3/zcBdrXDz+iPFBjAwyI0p8eO8fvb2WYVUqrnh66azjW/JpIoKjS00M+tJ/dR1115JE\nEhCFMyciWBbgUMyyFspaOyk77/2j8gnPZUQkUGaL/tmRnz/ZwzMua8iT26+mVowUMiPLvQec+U9p\nzdhgEMPDIsZF3/kCM5KQQFZogW9Newt/vath1+uW4FQQs8RZjEkOhxiMIMb+dqCt1A2XHVI7dmzg\n2SzKbPpvWv6a0fV9nndvzbKH2oGuFtb40VtrgJQRRGZkWXL31CePZPbkEIMYHhbQf9lHaUvThJg1\nwQUn1v6xfvxPJr7yl0ulIJyVUC89HTWypMyJnPV1UtY8Ay46Yax4Pl7x4w8C3/xX1uOMd7xsNCON\nzMiSZekje3Fh7ZE7IoaHQY6z730ndKSU2uC2PceHgb/8sJOzAEXBQL+6KGtLAqkzp84GZtB+wJ6/\nP3Dpw1NfM1aUDYSQsJbsz7Omvixf8+ierKUpxG5WOiPd+7CJYsSRGVmiVLz7hmYe3D2I4REDYHov\nPIXmNOuC2w5N4aG9gPbaxEQijTyjtROaZ1B2/8vSvlGlQlorrNh33XhP3G20hAWGB/YFvvr+X/wT\nZe2FoJCEQkY3NB2x9ASufm0urRVYlsXIITPyFO947U2H1yKGU4yzFz54ImXXvCUPCyYCzWmNQ+bI\nQDfwlX/s++VneUb7O/arFatYhSs+1NTNzE+PZojJrnkPZff3Hwx0KkQIpTbKrnrNdoUnvNNoMSLJ\njCyW44pv/udXP7SdxHCyIst//EgHv/iHOoiLJjKoTbQz6MI35ceOjQsXPN0/b+Gihu1654+feFRD\nwmoWcy9uouzpCazUf/PRlF34vlH33Pa5i5YWbZXcRlnL5yaAHRihZEYWi95vfwq6PjZRDCPHxPKS\npQvH7pSDEvGRe9/PM7o/tKMoc8mFgawu6c9GJTXCDqzkJy/qYtCicbIDsOwrbcB33zWW3ss+fvet\ni5SRtcPVD+ndLxMjmcxIk/WdOx341ol5hpGxZBEJxFBKictvvXUGg77eMGn7ieMCUErE2oxY5ZE9\ngNZOzvtoDRZlfbccDd89ZrR4fFcevHlefxdlP3nL8tyYYEvGgRFJZmTJAqVHvzUTek6uYRgZK4aY\nAI6JjdS3qBieXtbgXdM0FZgYHCxWM+AAmN65e0JjbWjm6nckVpaAf3Uk8KvDV4Q5S17VNXFZbydl\ntxxSB0QJjLCwGGlkRp5szn1H840P5Q0ObFGZgxOeywKMKFv84NL7/xNaalzo4vGdFEVMeHJH4Iz3\n/vXPU9/HKt89fPSEhJFPZiRadG523H7B2AmrGbGVZIlxYKXeb32aspYkX2rlwn+qJ0rLl8S4Cyt1\ntbDS1UfWixcDmRHI8cr3f/mkiYkDFltfRumeXepGC7L++5JDgeYZdLgdbj0sgf7bf3sana2s1NYB\nV06YNsYNeQEWI53MSOQfvB9+emQdFlufYd72X9nu8LEw/wZ0Ch0udjPogg+MBs3dnpU6+3sYdO1b\ngxhkxMgnMwJl4dp3AbcfErAYYhPYahY89YvJJzGkOeloTVsZct3hY4W19JtL21q6oDUpuAdo/8x4\nsYoRI53MSBTn/+RjcOMbElkMcd/iMCmJBLaC3ktWnD6jmSGdIo1NfPGNr+bO/XNGlG59I60hECOd\nDLrzwMAQixcDmRFp4NKPw69em8TAkOg7D+Pu/WJSyrHF+cF9oKONzhgzakIaCs1w5fbTdkwBa9l/\nn06H1NfDoK8/OeXEceLFRGZEivd9v2f6h/dIsSiLTp7Ymd8dkmCxxfV+rZEvFpo7QkYp1qZpoZmy\n//5onrLF2c8+QEu+6C6gq3jiDv25sQHxIiIzEkUV5/zx2K737SWLsuiw7LzG3x2UymKL67/yg8zI\nEsUinXRDE4Mue1+KPe/cbqA1tAPX1+5R2DEHGPEiIjMSxRDjte/mV4fnGOIY+s7/r9+/MiVL2OL6\n/+cjNOUTmlnt+l3um7T/dir85tGPslr3EYeFQKDMiBcRmZHIpnj+5+GmNwYGZYH46KL98mmWsOWV\nHv72TNpU7AYu+Djw5Y/XLUpW1NX/6s2s8d131ucos4x4UZEZkWzf9ZtlTT943cQUI8oyJ5bYYrKE\nLAEM8bEfnk7ZGcfW5296qO3Lx00Ncxbv3znxk6x228SJ48SLlMyIZKv02F7AHQcHLMqsiMQWY2HF\ngGX7kV/9B/DUFLHiLw+Ove9VCxryr2WV5kPqpu6dFy9aMiOSLd11EGV/3C9lJSO2HFsYCQzKli+6\npX//Qx6em5XuOxnOPI2r7mlj0Ne2P6IhJikvYjIjkh24/TAG3XFQghFb3Px0nBBEJMu9GnXfAaxx\n1V0dcP6xuVGqEWUG8eIkMyLZgTnfaGlMuuC6QyeILc5PT+7550lJKJbyJJGAS0/szqDmkHXT8qGF\ndzz9+ldMEYMsKxJ4cZIZgSwjitcf01w70AXNn5kkLLaseVOaPrbDsqf/NGX/eqzlYenPnmiiySSe\nAdx+SKkvG5uIQY5BvHjJjFil7/xrY71L7dB08nhZFltS//31L+v/5ue5/JhRxKX/M/F4oAXHpIuy\nheN5yZAZsXzvK+lMVWiDH70jbydsUY4KhZ+/60sfmqi+Xyz4CGVN3Qxp2+mQ/RLES4TMyLXikk83\njY0xa6XlC2OCxZaVBbl31o5jXbzmeNby7SXHjatJeemQGbni3PO7OmqUFdp5ekJ0wpZkZt+12265\nLIefvqCV1b53YG1DsSHB4iVDZsQynP+p9jpcaO3+XL3FlhWve9c5J0wRXnLxf7HKd3Y5eJQYZMXA\nS4TMyBV1wSc784rF1jM/U4sYYmSEATFcHAPxkRuP2DsP2WN7MOj8Cb1vnpSk4iVHZsQyXPDJGanD\nQGv7qXVZEINKQTEs8AQnWRDDwoiyYu/oHOau65oou/RdNa4PvBTJjGC+4JMzcjEpNHP3fg6IsoJS\nFk3g8R0VE4bRgsemPbl8YM+d9b0Pc9GTHdyzb+AlSmbEiuiCT3bmHXw6X/zYGFlA4RcrXjdx4SSe\nmBYYNv3L4ZfvO/8T8KV/Cec1X/6W5XOWHF4vKKW8BMmMWF4896ZPt9SmDivaeXCPwKDCje88+1/r\n/1y7aw6L4ZDxxF3vYbVz9nwHV7wnxJgzksVLkMyIVbzigzOX9bTVmelw26sDFnjRdTu+JpcpMMSy\nWMliNQuw2IAIQkDfz5d8mLV9M/e2CSArChAvPTIjVnbT0V0t0BncAvc0TKhBwIKF6ZRFk3ICTAwy\niLIoWaxkOQbE84ghhmgnsCTr690HaO2k7Myp+8+ZHHetrRcvaTIj1sBl//ql3Y5jyM03dt666x/2\n3lWP7QldLec0ZDu8ZhQWz3ApxxAjXpBhBbO/08UzLjlmdBJRykudzIhVuu6Y2/b5xXEMeurKTz/4\n+FvP+eDYS/+dVW7abxKyWCUKsTGWX1Za1MQzLvrHKYIYeMmTGbHiovm75pbPeepPu8561X7zizvO\n+tlh++buOpQ1Ht354cVjHpu6OLN23jnmU1YpFSFXxLWB9SkVk5zw05M7+meyytnsdNR4YQeiEC9p\nMiNXJGSB0vLa/rp8FHHpmFR9f3rsBFb5+WHnDbiDIbf8bt/XjaZsQckP3l3KdvgnuHu/wLpKi0bf\n+X9T9xs/ZWzh17e2scqt+5fS0YqSTeClTmZEs1hHsTD7b3P/2kjZbftsx2rf/vCMf6jDpf638Yzb\na1ZEWYCRPGavXPF3r7/g45Q1nzLm/v0Y9J1dRjXslLKKxUuezIuOC4WB/vuXbN//qlG339gBXH4S\nLcc/2n8SL+T+PX77elZZMH7xRaefO238Xg05gqgkMi9OAzGfJVlcEIUmzCI/NT65J6t8p+GdrNeX\nX3d9K6ssGO+nig015EWlkXmxsVgtWoGyKMulp/tKs5buOGfyzg3hwRsX71L6OM91Sd8ngbMPXrjL\nK2rtICqRzIuNZcQQI1YxZCkDWa6U5qC0KI7Wkz8+lXXd37BweZ/2ry8mOUWJMiMqi8yLjJFlhMHB\nsizKLIshFuuxeNmYUUmIQaxkYgJkClQamapKI1NVaWSqKo1MVaWRqao0MlWVRqaq0shUVRqZqkoj\nU1VpZKoqjUxVpZGpqjQyVZVGpqrSyFRVGpmqSiNTVWlkqiqNTFWlkamqNDJVlUamqtLIVFUamapK\nI1NVaWSqKo1MVaWRqao0MlWVRqaq0shUVRqZqkojU1VpZKoqjUxVpZGpqjQyVZVGpqrSyFRVGpmq\nSiNTVWlkqiqNTFWlkamqNDJVlUamqtLIVFUamapKI1NVaWSqKo1MVaWRqao0MlWVRqaq0shUVRqZ\nqkojU1VpZKoqjUxVpZGpqjQyVZVGpqrSyFRVGpmqSiNTVWlkqiqNTFWlkamqNDJVlUamqtLIVFUa\nmapKI1NVaWSqKo1MVaWRqao0MlWVRqaq0shUVRqZqkojU1VpZKoqjUxVpZGpqjQyVZVGpqrSyFRV\nGpmqSiNTVWlkqiqNTFWlkamqNDJVlUamqtLIVFUamapKI1NVaWSqKo1MVaWRqao0MlWVRqaq0shU\nVRqZqkojU1VpZKoqjUxVpZGpqjQyVZVGpqrSyFRVGpmqSiNTVWlkqiqNTFWlkamqNDJVlUamqtLI\nVFUamapKI1NVaWSqKo1MVaWRqao0MlWVRqaq0shUVRqZqkojU1VpZKoqjUxVpZGpqjQyVZVGpqrS\nyFRVGpmqSiNTVWlkqiqNTFWlkamqNDJVleb/A+xlIyo50TxwAAAAAElFTkSuQmCC\n'

image\_64\_decode = base64.decodebytes(b64\_file)

image\_result = open('MAP\_FILE.png', 'wb')

image\_result.write(image\_64\_decode)

MAP\_FILE = 'MAP\_FILE.png'

#Листинг 1.2. Определение класса Search и метода init ()

SA1\_CORNERS = (130, 265, 180, 315) # (UL-X, UL-Y, LR-X, LR-Y)

SA2\_CORNERS = (80, 255, 130, 305) # (UL-X, UL-Y, LR-X, LR-Y)

SA3\_CORNERS = (105, 205, 155, 255) # (UL-X, UL-Y, LR-X, LR-Y)

class Search():

"""Байесовская поисково-спасательная игра с 3 областями поиска."""

def \_\_init\_\_(self, name):

self.name = name

self.img = cv.imread(MAP\_FILE, cv.IMREAD\_COLOR)

if self.img is None:

print('Could not load map file {}'.format(MAP\_FILE),

file=sys.stderr)

sys.exit(1)

# Устанавливаем заполнители для фактического местоположения моряка

self.area\_actual = 0

self.sailor\_actual = [0, 0]

# Создаем массивы numpy для каждой области поиска, индексируя массив изображений.

self.sa1 = self.img[SA1\_CORNERS[1] : SA1\_CORNERS[3],

SA1\_CORNERS[0] : SA1\_CORNERS[2]]

self.sa2 = self.img[SA2\_CORNERS[1] : SA2\_CORNERS[3],

SA2\_CORNERS[0] : SA2\_CORNERS[2]]

self.sa3 = self.img[SA3\_CORNERS[1] : SA3\_CORNERS[3],

SA3\_CORNERS[0] : SA3\_CORNERS[2]]

# Устанавливаем начальные целевые вероятности для каждой области поиска моряка

self.p1 = 0.2

self.p2 = 0.5

self.p3 = 0.3

# Инициализируем вероятности эффективности поиска.

self.sep1 = 0

self.sep2 = 0

self.sep3 = 0

#Листинг 1.3. Определение метода для отображения базовой карты

def draw\_map(self, last\_known):

"""Отобразите базовую карту с масштабом, последним известным местоположением xy, областями поиска."""

# Рисуем масштабную линейку

cv.line(self.img, (20, 370), (70, 370), (0, 0, 0), 2)

cv.putText(self.img, '0', (8, 370), cv.FONT\_HERSHEY\_PLAIN, 1, (0, 0, 0))

cv.putText(self.img, '50 Nautical Miles', (71, 370),

cv.FONT\_HERSHEY\_PLAIN, 1, (0, 0, 0))

# Рисуем и нумеруем области поиска.

cv.rectangle(self.img, (SA1\_CORNERS[0], SA1\_CORNERS[1]),

(SA1\_CORNERS[2], SA1\_CORNERS[3]), (0, 0, 0), 1)

cv.putText(self.img, '1',

(SA1\_CORNERS[0] + 3, SA1\_CORNERS[1] + 15),

cv.FONT\_HERSHEY\_PLAIN, 1, 0)

cv.rectangle(self.img, (SA2\_CORNERS[0], SA2\_CORNERS[1]),

(SA2\_CORNERS[2], SA2\_CORNERS[3]), (0, 0, 0), 1)

cv.putText(self.img, '2',

(SA2\_CORNERS[0] + 3, SA2\_CORNERS[1] + 15),

cv.FONT\_HERSHEY\_PLAIN, 1, 0)

cv.rectangle(self.img, (SA3\_CORNERS[0], SA3\_CORNERS[1]),

(SA3\_CORNERS[2], SA3\_CORNERS[3]), (0, 0, 0), 1)

cv.putText(self.img, '3',

(SA3\_CORNERS[0] + 3, SA3\_CORNERS[1] + 15),

cv.FONT\_HERSHEY\_PLAIN, 1, 0)

# Укажем последнее известное местонахождение моряка.

cv.putText(self.img, '+', (last\_known),

cv.FONT\_HERSHEY\_PLAIN, 1, (0, 0, 255))

cv.putText(self.img, '+ = Last Known Position', (274, 355),

cv.FONT\_HERSHEY\_PLAIN, 1, (0, 0, 255))

cv.putText(self.img, '\* = Actual Position', (275, 370),

cv.FONT\_HERSHEY\_PLAIN, 1, (255, 0, 0))

cv.imshow('Search Area', self.img)

cv.moveWindow('Search Area', 750, 10)

cv.waitKey(500)

#Листинг 1.4. Определение метода для случайного выбора фактического местоположения моряка

def sailor\_final\_location(self, num\_search\_areas):

"""Верните фактическое местоположение x,y пропавшего моряка."""

# Найдем координаты моряка относительно любого подмассива области поиска.

self.sailor\_actual[0] = np.random.choice(self.sa1.shape[1])

self.sailor\_actual[1] = np.random.choice(self.sa1.shape[0])

# Выберем область поиска наугад.

area = int(random.triangular(1, num\_search\_areas + 1))

# Преобразуем координаты локальной области поиска в координаты на карте.

if area == 1:

x = self.sailor\_actual[0] + SA1\_CORNERS[0]

y = self.sailor\_actual[1] + SA1\_CORNERS[1]

self.area\_actual = 1

elif area == 2:

x = self.sailor\_actual[0] + SA2\_CORNERS[0]

y = self.sailor\_actual[1] + SA2\_CORNERS[1]

self.area\_actual = 2

elif area == 3:

x = self.sailor\_actual[0] + SA3\_CORNERS[0]

y = self.sailor\_actual[1] + SA3\_CORNERS[1]

self.area\_actual = 3

return x, y

#Листинг 1.5. Определение методов для случайного выбора эффективности поиска и выполнения самого поиска

def calc\_search\_effectiveness(self):

"""Установите десятичное значение эффективности поиска для каждой области поиска."""

self.sep1 = random.uniform(0.2, 0.9)

self.sep2 = random.uniform(0.2, 0.9)

self.sep3 = random.uniform(0.2, 0.9)

def conduct\_search(self, area\_num, area\_array, effectiveness\_prob):

"""Возвращает результаты поиска и список искомых координат."""

local\_y\_range = range(area\_array.shape[0])

local\_x\_range = range(area\_array.shape[1])

coords = list(itertools.product(local\_x\_range, local\_y\_range))

random.shuffle(coords)

coords = coords[:int((len(coords) \* effectiveness\_prob))]

loc\_actual = (self.sailor\_actual[0], self.sailor\_actual[1])

if area\_num == self.area\_actual and loc\_actual in coords:

return 'Found in Area {}.'.format(area\_num), coords

return 'Not Found', coords

#Листинг 1.6. Определение способов применения теоремы Байеса и отрисовка меню в оболочке Python

def revise\_target\_probs(self):

"""Обновлять вероятности попадания в целевую область на основе эффективности поиска."""

denom = self.p1 \* (1 - self.sep1) + self.p2 \* (1 - self.sep2) \

+ self.p3 \* (1 - self.sep3)

self.p1 = self.p1 \* (1 - self.sep1) / denom

self.p2 = self.p2 \* (1 - self.sep2) / denom

self.p3 = self.p3 \* (1 - self.sep3) / denom

def draw\_menu(search\_num):

"""Распечатайте меню вариантов для проведения поиска по области."""

print('\nSearch {}'.format(search\_num))

print(

"""

Choose next areas to search:

0 - Quit

1 - Search Area 1 twice

2 - Search Area 2 twice

3 - Search Area 3 twice

4 - Search Areas 1 & 2

5 - Search Areas 1 & 3

6 - Search Areas 2 & 3

7 - Start Over

"""

)

#Листинг 1.7. Определение начала функции main(), используемой для запуска программы

def main():

app = Search('Cape\_Python')

app.draw\_map(last\_known=(160, 290))

sailor\_x, sailor\_y = app.sailor\_final\_location(num\_search\_areas=3)

print("-" \* 65)

print("\nInitial Target (P) Probabilities:")

print("P1 = {:.3f}, P2 = {:.3f}, P3 = {:.3f}".format(app.p1, app.p2, app.p3))

search\_num = 1

#Листинг 1.8. Использование цикла для выбора пунктов меню и запуска игры

while True:

app.calc\_search\_effectiveness()

draw\_menu(search\_num)

choice = input("Choice: ")

if choice == "0":

sys.exit()

elif choice == "1":

results\_1, coords\_1 = app.conduct\_search(1, app.sa1, app.sep1)

results\_2, coords\_2 = app.conduct\_search(1, app.sa1, app.sep1)

app.sep1 = (len(set(coords\_1 + coords\_2))) / (len(app.sa1)\*\*2)

app.sep2 = 0

app.sep3 = 0

elif choice == "2":

results\_1, coords\_1 = app.conduct\_search(2, app.sa2, app.sep2)

results\_2, coords\_2 = app.conduct\_search(2, app.sa2, app.sep2)

app.sep1 = 0

app.sep2 = (len(set(coords\_1 + coords\_2))) / (len(app.sa2)\*\*2)

app.sep3 = 0

elif choice == "3":

results\_1, coords\_1 = app.conduct\_search(3, app.sa3, app.sep3)

results\_2, coords\_2 = app.conduct\_search(3, app.sa3, app.sep3)

app.sep1 = 0

app.sep2 = 0

app.sep3 = (len(set(coords\_1 + coords\_2))) / (len(app.sa3)\*\*2)

elif choice == "4":

results\_1, coords\_1 = app.conduct\_search(1, app.sa1, app.sep1)

results\_2, coords\_2 = app.conduct\_search(2, app.sa2, app.sep2)

app.sep3 = 0

elif choice == "5":

results\_1, coords\_1 = app.conduct\_search(1, app.sa1, app.sep1)

results\_2, coords\_2 = app.conduct\_search(3, app.sa3, app.sep3)

app.sep2 = 0

elif choice == "6":

results\_1, coords\_1 = app.conduct\_search(2, app.sa2, app.sep2)

results\_2, coords\_2 = app.conduct\_search(3, app.sa3, app.sep3)

app.sep1 = 0

elif choice == "7":

main()

else:

print("\nSorry, but that isn't a valid choice.", file=sys.stderr)

continue

#Листинг 1.9. Завершение и вызов функции main()

app.revise\_target\_probs() #Используем правило Байеса для обновления целевых вероятностей.

print("\nSearch {} Results 1 = {}"

.format(search\_num, results\_1), file=sys.stderr)

print("Search {} Results 2 = {}\n"

.format(search\_num, results\_2), file=sys.stderr)

print("Search {} Effectiveness (E):".format(search\_num))

print("E1 = {:.3f}, E2 = {:.3f}, E3 = {:.3f}"

.format(app.sep1, app.sep2, app.sep3))

# Выведем вероятности цели, если моряк не найден, в противном случае покажем местоположение.

if results\_1 == 'Not Found' and results\_2 == 'Not Found':

print("\nNew Target Probabilities (P) for Search {}:"

.format(search\_num + 1))

print("P1 = {:.3f}, P2 = {:.3f}, P3 = {:.3f}"

.format(app.p1, app.p2, app.p3))

else:

cv.circle(app.img, (sailor\_x, sailor\_y), 3, (255, 0, 0), -1)

cv.imshow('Search Area', app.img)

cv.waitKey(1500)

main()

search\_num += 1

if \_\_name\_\_ == '\_\_main\_\_':

main()