**데이터기반 행정으로 국민의 삶의질을 개선하라!**

**데이턴십 해커톤 제 4회**

|  |
| --- |
| **서울시의 유기동물 보호 및 관리를 위한  공공 동물보호센터 최적입지 선정 분석** |

분석 결과보고서

|  |  |
| --- | --- |
| 참여조 : | 48조 |
| 참여자 : | 팽대원(조장) |
|  | 강예지 |
|  | 김하영 |
|  | 문광표 |
|  | 이예림 |
|  |  |



Copyright ⓒ CSLEE Consortium

CSLEE Consortium의 사전 승인 없이 본 내용의 전부 또는 일부에 대한 복사, 배포, 사용을 금합니다.

**목 차**

**1 분석 개요 4**

**1.1 분석 배경 및 개요 4**

**1.2 분석 목적 및 방향 7**

**1.3 분석 결과 활용 방안 7**

**2 분석 데이터 8**

**2.1 분석 데이터 목록 8**

**2.2 데이터 상세 설명 9**

**2.3 데이터 정제 방안 12**

**3 분석 프로세스 13**

**3.1 분석 프로세스 13**

**3.2 분석 내용 및 방법 14**

**4 분석결과 35**

**4.1 AHP 분석 35**

**3.2 군집화 36**

**3.2 구별 우선 순위 시각화 37**

**3.2 최종 입지 선정 시각화 39**

**5 활용 방안 45**

**5.1 문제점 개선 방안 .............................................................45**

**5.2 업무 활용 방안 ................................................................45**

**6 참고자료** **46**

**7 부록** **47**

# 분석개요

## 분석 배경 및 개요

### 국내 반려동물 현황



그림 1 국내 반려동물 양육가구 현황 그림 2 반려동물 연관산업 규모 전망

* ‘2021 한국반려동물보고서’에 따르면, 20년도 말 기준 한국의 반려동물 가구는 604만 가구로 전체 가구의 29.7%이며 반려인은 1,448만 명으로 집계됨[[1]](#footnote-2) (황원경 & 손광표, 2021)
* 지역별 반려가구 현황을 보면 서울이 131만 가구, 경기/인천이 196만 가구로 전체 반려가구의 절반 이상을 차지하며, 전체 가구 중 반려 가구의 비중이 서울이 33.7%, 경기/인천 32.5%로 수도권이 비수도권(26.5%)와 비교해 수도권이 비수도권(26.5%)와 비교해 높은 비율을 차지하고 있음
* 반려동물의 수와 반려동물을 기르는 가구는 해마다 증가하고 있으며, 한국농촌경제연구원에 따르면 관련 사업도 더불어 증가하며 2027년도에는 약 6조원 규모의 시장이 형성될 것으로 예상됨 (지인배, 김현중, 김원태, 서강철, 2017)

### 국내 유기동물 발생 현황 및 문제점

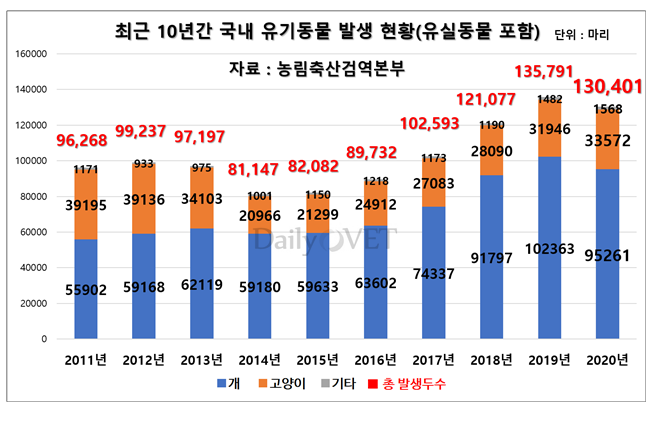


그림 3 최근 10년간 국내 유기동물 발생 현황(유실동물 포함)

* 반려동물의 수가 증가함에 따라 동물에 대한 관심은 늘어났으나 이와 더불어 유기동물의 발생 건 수가 증가하며 심각한 상황임
* 국내 유기동물 발생 수는 14년 이후로 꾸준하게 증가하였고, 20년도에 전년대비 3.9% 감소하였으나, 유기동물 발생 수가 약 13만 마리로 동물보호센터 수용 능력에 비해 압도적으로 많음
* 유기동물 수에 비해 이들을 보호, 조치할 수 있는 동물보호센터의 수용 규모가 부족하여 위생 문제 등 관리의 어려움이 빈번하게 발생
* 어려운 관리 문제 등으로 인해 유기동물 중 다른 가정으로 분양되는 동물의 수에 비해 자연사 및 안락사의 비율이 높음 [[2]](#footnote-3)
* 전체 동물보호센터의 유기동물 일시수용 가능 개체 수와 평균 보호기간을 고려하면 현재 수용능력을 초과하며, 전문가들은 현재와 같은 수용 규모로는 유기동물에 대한 적절한 보호 조치에 한계가 있을 것이라고 지적함

### 국내 동물보호센터의 유실 ·유기 동물 구조 ·보호 현황

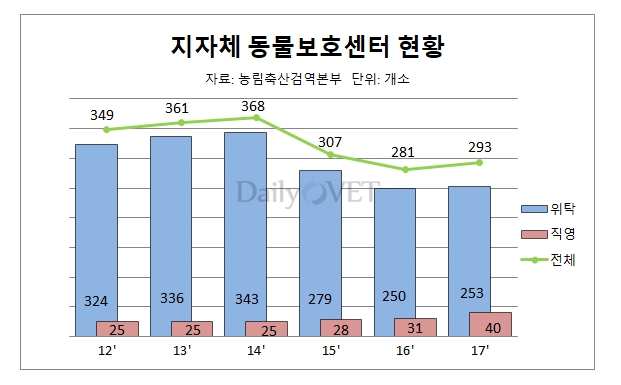


그림 4 지자체 동물 보호센터 현황

* 2014년 이후로 전체 지자체 동물 보호센터의 수는 감소하는 추세이며 직영 동물보호센터 수는 소폭 증가하였지만, 위탁 동물보호센터 수는 대폭 감소함
* 2020년 말 기준으로 동물보호센터는 280개소이며, 지방 자치단체가 운영하는 동물 보호센터 중 직영 동물 보호센터가 차지하는 비중이 낮음[[3]](#footnote-4) (농림축산검역본부 동물보호과, 2021)
* 사설 동물 보호시설은 「동물 보호법」의 적용 대상이 아니기 때문에 지방자치단체가 운영하는 보호센터에 적용이 되는 보호동물에 대한 인도적 처리, 즉 안락사의 대상, 원칙, 절차 및 방법에 대한 규정이 적용되지 않음
* 전국 사설 보호소 83곳 중 60곳은 ‘국토의 계획 및 이용에 관한 법률’, ‘수도법’, ‘수질 및 수생태계 보전에 관한 법률’ 등의 법률을 위반함 (이혜원, 19년 3월)
* 현행 동물 보호법 15조에 따르면 유기동물의 구조•보호에 관해 동물보호센터를 설치•운영할 수 있는 주체는 시•도지사 또는 시장•군수•구청장과 이들이 지정하는 기관이나 단체에 한정되어 지방자치단체와 무관한 민간인이 운영하는 사설보호소에 대한 규정은 전혀 없고 관리가 어려움

### 분석 개요

* 반려동물에 대한 관심이 늘어나고 있고 반려동물과 함께 사는 가구의 수가 늘어나고 있음. 특히 서울시의 반려동물의 개체 수는 전국적으로 높음. 하지만 유기동물 발생 수는 전국적으로 약 13만 마리로 심각한 상황
* 증가하는 유기동물의 수에 따라 보호소의 과포화, 과밀화 문제 등이 일어나고 있으며 일부 보호소에서 ‘파보 바이러스’ 등의 감염병이 돌아 많은 개체들이 죽는 경우가 발생
* 유실 · 유기동물의 경우 길에서의 생활로 감염병에 노출 위험이 높은 반면, 많은 보호소들이 예산과 공간 등의 문제로 새로 입소한 동물을 위한 격리공간을 갖추기 어려워 비슷한 유형의 사고가 반복해서 일어날 가능성이 있음
* 민간 운영 동물 보호센터의 경우 윤리적, 위생적 한계를 가지고 있으며 관리가 어려움. 또한 유실, 유기동물의 관리를 개인적인 차원에서 방치하다 보니 ‘캣맘 논란’ 등과 같은 다양한 부작용이 나타나고 있음
* 따라서 국가 운영 ‘공공 동물보호센터’의 추가 설립 추진 필요

## 분석 목적 및 방향

* 서울시 공공 유기동물 보호소 최적 입지 선정 결과를 활용한 시설 확충을 통한 유기동물의 위생 문제 해결 및 동물 복지 정책 도입
* 최적 입지 선정을 통해 유기동물 보호소 추가 건립 정책 효율 극대화
* 유기동물 관리 및 처리로 인한 생태계 위험 및 사회적 비용 감소
* 유기동물 입양 접근성 향상을 고려한 최적 입지 선정을 통해 인간과 동물의 공생 실현

## 분석 결과 활용 방안

* (업무 활용) 신규 공공 유기동물 보호소 최적 입지 선정 시에 후보군 선정에 활용, 기존 선정된 입지에 대한 활용도 제고 및 확대 운영 방안 정책에 활용
* (정책 마련) 정부의 국정운영 5개년 계획인 ‘사람과 동물이 공생하는 국토 환경 조성’ 동물 복지 분야와 관련하여 ‘유기동물 입양 활성화’ 정책 마련
* (확대 적용) 분석 시 사용된 변수를 활용하여 ‘반려동물 지원센터 설치’ 및 ‘길고양이 중성화 사업 지원’ 정책에 확대 적용 가능

# 분석 데이터

## 분석 데이터 목록

표 1 분석 데이터 목록

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 분석 데이터 | 기간 | 제공기관 |
| 참여성 | 서울시 가구원수별 가구수 | 21.08 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 서울시 국민기초생활보장 일반수급자 성별, 연령별 현황 | 21.01 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 서울시 직업유형별 취업인구 | 17.07 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 서울시 자치구별 월소득 통계 | 16.09 | 신한은행  (https://www.shinhan.com/index.jsp) |
| 서울시 주택종류별 주택 통계 | 21.08 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 유기  동물 | 유기동물 분석 보고서 | 16.01~  20.12 | 동물 자유 연대  (https://www.animals.or.kr/) |
| 서울시 유기동물 보호 현황 | 21.03 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 서울시 반려동물 유무 및 취득경로 | 21.03 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 접근성 | 전국 행정구역 (시군구) | 21.01 | 공간정보시스템 – 지오서비스  (http://www.gisdeveloper.co.kr/) |
| 전국 행정구역 (읍면동) | 21.01 | 공간정보시스템 – 지오서비스  (http://www.gisdeveloper.co.kr/) |
| 서울시 공동주택 가격 및 공간 정보 | 21.07 | 국토교통부 – 공간정보포털  (http://www.nsdi.go.kr/lxportal/?menuno=2679) |
| 서울시 학교 기본 정보 | 21.06 | 나이스  (https://open.neis.go.kr/portal/mainPage.do) |
| 서울시 주민자치센터 정보 | 16.02 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 서울시 소방서 관할 위치정보 | 17.07 | 서울 열린 데이터 광장  (https://data.seoul.go.kr/) |
| 서울시 동물보호센터 위치 | - | 동물 보호 관리 시스템  (https://www.animal.go.kr/front/index.do) |
| 설문  조사 | (일반인 대상)  입지 선정 요인 중요도 설문조사 | 21.08 | 온라인 설문조사 |

## 데이터 상세 설명

표 2 데이터 상세 설명

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 분석 데이터 | 데이터 형식 | 생성주기 |
| 공공 데이터 | 서울시 가구원수별 가구수 | CSV | 5년 |
| 서울시 국민기초생활보장 일반수급자 성별, 연령별 현황 | CSV | 1년 |
| 서울시 직업 유형별 취업 인구 | CSV | 5년 |
| 서울시 자치구별 월소득 통계 | CSV | 1년 |
| 서울시 주택 종류별 주택 통계 | CSV | 5년 |
| 서울시 유기동물 보호 현황 | CSV | 1년 |
| 서울시 반려동물 유무 및 취득 경로 | CSV | 1년 |
| 서울시 공동주택 가격 및 공간 정보 | CSV | 반기 |
| 서울시 학교 기본 정보 | CSV | 실시간 |
| 서울시 소방서 관할 위치정보 | CSV | 수시 |
| 서울시 주민자치센터 정보 | CSV | 1회 |
| 민간 데이터 | 유기동물 분석 보고서 | PDF | 5년 |
| (일반인 대상) 입지 선정 요인 중요도 설문조사 | CSV | 1회 |
| 서울시 동물보호센터 위치 | CSV | 1회 |
| 전국 행정구역 (시군구) | SHP | 실시간 |

1. 서울시 가구원별 가구수[.CSV]

* 서울시 열린 데이터 광장 제공, 구별 가구원 수(1인 ~ 7인이상)

테이블이(가) 표시된 사진

자동 생성된 설명

그림 5서울시 구별 가구원 수

1. 서울시 국민기초생활보장 일반수급자 성별, 연령별 현황[.CSV]

* 서울시 열린 데이터 광장 제공, 행정동 별 연령별(0~100세)의 인구 수에 대한 정보가 담긴 데이터

테이블이(가) 표시된 사진

자동 생성된 설명

그림 6 서울시 국민기초생활보장 일반수급자 성별, 연령별 현황

1. 서울시 직업 유형별 취업 인구[.CSV]

* 서울시 열린 데이터 광장 제공, 직업 유형별(관리자, 전문가, 사무종사자, 농림어업 종사자 등) 취업 인구 데이터

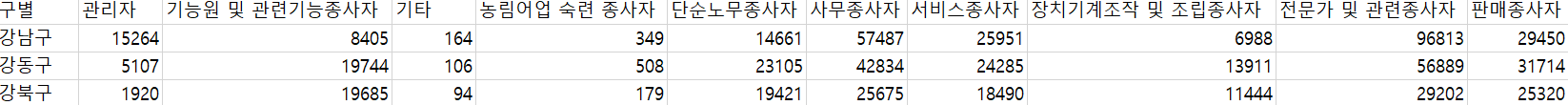


그림 7 서울시 직업 유형별 취업 인구

1. 서울시 자치구별 월 소득 통계[.CSV]

* 신한은행 제공, 구별 세전 월 소득 데이터

테이블이(가) 표시된 사진

자동 생성된 설명

그림 8 구별 세전 월 소득 데이터

1. 서울시 주택 종류별 주택 통계[.csv]

* 서울시 열린 데이터 광장에서 제공, 구별 주택 종류(아파트, 단독, 연립, 다세대, 비거주용 주택) 개수

![테이블이(가) 표시된 사진

자동 생성된 설명](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAeAB4AAD/4RDaRXhpZgAATU0AKgAAAAgABAE7AAIAAAAFAAAISodpAAQAAAABAAAIUJydAAEAAAAKAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHVzZXIAAAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAM1MQAAkpIAAgAAAAM1MQAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAyMTowODoxNiAxNTo1MTo0NwAyMDIxOjA4OjE2IDE1OjUxOjQ3AAAAdQBzAGUAcgAAAP/hCxdodHRwOi8vbnMuYWRvYmUuY29tL3hhcC8xLjAvADw/eHBhY2tldCBiZWdpbj0n77u/JyBpZD0nVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkJz8+DQo8eDp4bXBtZXRhIHhtbG5zOng9ImFkb2JlOm5zOm1ldGEvIj48cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiPjxyZGY6RGVzY3JpcHRpb24gcmRmOmFib3V0PSJ1dWlkOmZhZjViZGQ1LWJhM2QtMTFkYS1hZDMxLWQzM2Q3NTE4MmYxYiIgeG1sbnM6ZGM9Imh0dHA6Ly9wdXJsLm9yZy9kYy9lbGVtZW50cy8xLjEvIi8+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDIxLTA4LTE2VDE1OjUxOjQ3LjUxMzwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iPjxkYzpjcmVhdG9yPjxyZGY6U2VxIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpsaT51c2VyPC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIAHcBwgMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APfbvTtAsLOW7vrPTba2hUvLNNFGiIo6ksRgD3NYH/CVfDH/AKD3hL/wMtv8a6LV9ctNEWJryK/kEpIX7Fp090Rj1ESNt698ZrFPxH8Pm8ayEetG7EXm+R/wj9/v2Zxu2+TnGeM9KTYzVsrHw9qVlHeada6Zd2sq7o54I43Rx6hhwRUfleF/7J/tTy9I/s/Z5n2vbF5W3+9v6Y981h/Cq4J+HmnafPZX9leWduFuIrywmt9rsWJCmRQG5z90nFefHwp44Pwq/wCEZ+xeKftP2QQfZvO0gWec9N4PnBPfO6qeja/rqJHrcaeGpddm0eOzsmv4bdLmSH7IOI2YqrbtuDkqeM54q9/Yek/9Auy/8B0/wrgvBGg6tpPxQ1efU9OmtoZNJt0SU6pc6jGzCSQlRPOqnIyPkHAyD3r0ujon6/mxdX8vyRR/sPSf+gXZf+A6f4Uf2HpP/QLsv/AdP8KvUUhmWuiaV9skX+zLPAjQgfZ19W9qx7zXvh3p95LaX+q+GLW5hbbJDPc26Oh9CpOQa6Zf+P6X/rmn82rg/DWuQ+HL/wAS2+rafrSPPrdxcRNBol5OkkbBAGDxxMpBwe9C3t5fqv8AMOl/66nQxzeEprnToLeLTZ21RHks2ggWRJ1QAsQ6grgAjvz2qTVR4T0G2W41waNpsDvsWW88qFWbGcAtgZwDx7Vy/iK7u9Y8c+C7nw9OLKWSO/2vqemzDACICDCzROOnBJHryKk+IOtabDq2ieX4g8N2+paPfi7ls9V1ZbQsphkT+67A/vAeV6UPp/XUEaMXiX4bTzJDBrXhWSWRgqIl3bFmJ4AAB5NbGo2vhrSLF73VoNKsbWPG+e5SONFycDLNgDk4rhNL+MqajNcCWfwZZpbXTW7NP4s2mQLjMkY+z/MpzwcjOD0rrtZ8a6TZaLa39jrnh8xXjlYLi+1VIIZFHDFHAbeQcfKMfUUPa4dbFH/hKvhj/wBB7wl/4GW3+NbFjF4W1PTf7R01NIu7HDH7TbiKSLC9fmHHGDmuIj1LwiurN4gk+JmhP4jK+WLr7dD9nWH/AJ4CDzf9Xn5vvbywzu7V09l4ih8UeFtXS0vtH1C4igkQro2oC7Byh2k4UFSSDheenU1MnaLa3Q46ySFsNa+H2q3sdnpepeGr26kzsgtp7eSR8DJwqkk8An8Kk0i/8H67LBHpUFlO9xaC9iH2LbuhLFQ3zKMcgjB59q5n4d3j2Gj6LY32p+LJJ1sYoGsL3QHit4HCAEeaLRSApBALSEepNYfwatpbfWdLaW9nuhL4VidElWMCEfaH+Rdqgkf72489a0sufl/raT/REJ+7zen4tf5nrP8AYek/9Auy/wDAdP8ACsK2134eXl9HZWmqeGZ7uR/LSCK4t2kdum0KDkn2rrK878d6naaj4o03SZNdXRrTR3Go6lqPmQoLVyClvGWlDIGdmLYIJwue4NR1RXRm7a3/AIPvdQisraCye4mnuLeNPsWMyQHEoyVx8p79D2zWz/Yek/8AQLsv/AdP8K8i8CyRzeKvD8kF9/aMT6rrzJebkb7QN64fKAKc9flAHPAr2uq+yn3SDq/V/mZa6JpX2yRf7Ms8CNCB9nX1b2qnfS+D9Lv4LHU30Ozu7nHkW9wYY5JcnA2qcE88cVtL/wAf0v8A1zT+bVxPg7TrLWdJ8Vy6xBHcSajq15b3omG793GxjjjOf4QijA6ck96m9r+Sv+X+YdPnb8/8joZ4fDltq9ppc1jZLeXkckkEX2QHese3ec7cDG9ep5zxVPU9U8B6JeG01m+8O6fchQxhu5oInweh2sQcVwejazf+d8Nb82F9rFy+j3q7IJIhLIo8gByZXRTkAE8556V3HiC8uYfHfg+OO4mgguZbpJoA5AkP2csoYDg4IJ56HpVNW/H82gW2pGPEHw9ksbi7tb7QbyG12eebPyrgx72CLlY8kZYgdK6H+w9J/wCgXZf+A6f4V5jrd/d3fwh8VS3Us13JD4gmijVny2xL5QqLuOAAAABkAV0nxDvJ774L+JZ7rTrnTZDp04NvdtEzj5TyTG7rg/71Tf3ebyv96T/UcVeXL52/Gx1X9h6T/wBAuy/8B0/wo/sPSf8AoF2X/gOn+FePR2Pgjykz4h+EOdoznQYc/wDpZXrfhr7APDViNHl06azWICOTS41S2bHBMaqzBVznjJx6mqsQnexN/Yek/wDQLsv/AAHT/Cj+w9J/6Bdl/wCA6f4VeopFGXbaJpTREtplmT5jjm3X+8faqS3Hg1tZOkLLoR1MdbIND5w/4B979K05kuJNHu0sXEd0wmELnor5bafzxXlv2rQrj4OJ4SsY0/4SdbQRx6SpC3sd+B/rSpwykSfP5pwMfNuxzSvv/V/6/VDtt/X9f8OeirD4cfW5NIWxsjfRW63LxfZBxGzMobO3HVWGM54qjqOseANIvpLLVtR8N2N3HjfBcz28ci5GRlWIIyCD+Ncx4FtpLT4p6jHPYavZS/2Fbb11a/8AtcjnzpcsredNtU/3dwwc8c5N7Vv+E0/4SzVtn/CS/wBlb4/7P/sf+ytmzyl37vtPz537van0Xz/BtCWrfy/JM33uvB6xaZKqaZNFq03kWUsECypO+1mwGUEdFbknHFO1e48GeHzENem0LTDNnyhetDD5mMZxuxnGR09a4/8AsW7tbLwLp9noes20Gla0C5v/ACJZPL8mUmRzbsyBdzYydvPaui8T+MbXTdSbS18TeGtGk8sGSbUb9fPhJ5GLclQcjBDFx1+6R1Ht8/wsn+of5fq0R/8ACVfDL/oPeE//AAMtv8a2NQi8LaRp/wBu1VNHsbTIH2i5EUcfPT5mwOa4rTdY8IeFVlk8L+PtAna4czXkGp6xEwu5j96XzA2Y3buQrLgABB1q/wCO7lvEHgPSL/SZnuV/tS0ne40Mfb/KVJRvePajb9uDzsPTkdqO3qvx/r/Owf8ABNRdd8AS2E97aXmhXlvbvFHM9l5dx5bSOEQMI8kZY4/P0NaWnQeHNWW5bT7GymFrcPazf6IF2yIcMvKjOPUcVwXi2+W9+HOo2iav4ivJjeWLC41bRzZtEDdxD5CbaJGOecEMfwrovhhC9vpevQzXMt06a/eK08wUPId4+YhFVcn2AHtTWt/6/l/zYntfz/R/5HRXWiaUtnMy6ZZgiNiCLdeOPpUOpxeF9Es/tesx6Rp9tuC+ddrFEmT0G5sDNal5/wAeM/8A1zb+Vcr8RP8AV+Gf+xisv/QjS6pd2l97sPo32T/BFnTNU8B61d/ZdGvvDuoXO0v5NpLBK+0dTtUk4q7pMHhzXNJttT0uxsp7O6QSQyfZAu5T3wygj8RWy/3G+lcH8Hb+5uPhvo1vNpN5axQ2aeXdTPCY7jk/cCSM4/4Eq0LW/wAv1/yA6iLT/D89xPBBaabLNbMFnjSKMtESAwDADIJBB57Gs7SL/wAH67PFDpUFlPJNbfa0H2LbmLeU3fMo/iUjHXjpXKeFvEV3e+N/Gv8AYFla3MtxdxS232y7aCK4hiT7PI6Osbk7ZIiPu45HPTOf8I9x1rSt4Ab/AIRgZAOQD9sl70R1a81/7a3+aB6J91/ml+p2t1rvw8sb6SyvtU8M211E2ySCa4t0dG9CpOQfapX1DwdHevaPBZCdL1NPZfsXSd0EipnbjlSDnp75qt8Pv9Z4r/7GK6/9Bjrhb+6mHxKubUWFw0LeMrJzeBo/KU/YlGwjdv3d+Fx79qI6uK7pP73H/MJaKT7X/C/+R6Tq9x4M8PmIa9NoWmGbPlC9aGHzMYzjdjOMjp61m/8ACVfDL/oPeE//AAMtv8ak8T+MbXTdSbS18TeGtGk8sGSbUb9fPhJ5GLclQcjBDFx1+6R157TdY8IeFVlk8L+PtAna4czXkGp6xEwu5j96XzA2Y3buQrLgABB1pJ9wZ2uoReFtI0/7dqqaPY2mQPtFyIo4+enzNgc1mrrvgCWwnvbS80K8t7d4o5nsvLuPLaRwiBhHkjLHH5+hrL8d3LeIPAekX+kzPcr/AGpaTvcaGPt/lKko3vHtRt+3B52HpyO1ZHi2+W9+HOo2iav4ivJjeWLC41bRzZtEDdxD5CbaJGOecEMfwpre3nb8v8w7f13O906Dw5qy3LafY2Uwtbh7Wb/RAu2RDhl5UZx6jip7rRNKWzmZdMswRGxBFuvHH0rnfhhC9vpevQzXMt06a/eK08wUPId4+YhFVcn2AHtXYXn/AB4z/wDXNv5UdE+6T+9B1a83+ZX/ALD0n/oF2X/gOn+FFXqKACuVSzuR8YJr028v2U6EkQn2HYX89jt3dM45x1xXSefJ/wA+sv5p/wDFUefJ/wA+sv5p/wDFULRp/wBbNfqHRr+t7k1eQeIrWO+8a61/anhrR1WKZFt7q58CXeqPdJ5SEuZ4mCnBJXHbbXrPnyf8+sv5p/8AFUefJ/z6y/mn/wAVStrcd9Dzv4d6JY2Piy/v7OOK2eeySE29n4TutHgAR2O8mUkO5344OcAelelVD58n/PrL+af/ABVHnyf8+sv5p/8AFVV9EietyaiofPk/59ZfzT/4qjz5P+fWX80/+KpDBf8Aj+l/65p/Nq5HVNauNF+JhkvYtYk0mTSFVBZ2FzdRCfzmzkRIwDbccnHGK6lZn+2SH7NLny04yvHLe9S+fJ/z6y/mn/xVHVP+trB0a/re5y8eo6J4g8XaTeJFr0V5YrOsAn0a7t4T5ijdveSEKOF4+Yc+uRW9q2m3WpRxrZ61faSUJLPZpAxk9j5sbj8sVa8+T/n1l/NP/iqPPk/59ZfzT/4qgDi/Eul+KNG0b7bo/ivXtSuo7m3X7K9pZuro0yLJkJbBsBCxyCMYz2ruqh8+T/n1l/NP/iqPPk/59ZfzT/4qjpYDm2s7n/hcKXv2eX7L/YLRefsPl7/tAO3d0zjnHXFdVUPnyf8APrL+af8AxVHnyf8APrL+af8AxVC0SX9bt/qHVv8AraxkXvg/Tb+9lup7nWkklOWWDXb2FB9ESUKv0AFZOj6Pc6L8SriG0l1iTSH0lWze31zdRCfziDhpXYBtuOAeldb58n/PrL+af/FUefJ/z6y/mn/xVC0af9bWB6q39bmdY+E/D+lXU13pGiadp13MjI91Z2kcUpDHJ+YLk8gHnuKd4d8OWPhnTWtNP81zJIZri4uJDJNcSt96SRz95jgfQAAYAAq/58n/AD6y/mn/AMVR58n/AD6y/mn/AMVRsB5tovh3VtN8daRLd2Eqxf2nrVyZFG9FjmYNGWZchdw6A4Nen1D58n/PrL+af/FUefJ/z6y/mn/xVHRLtoHVvuC/8f0v/XNP5tXPah4FtLy+vp7XVNT0yLU8HULWxlRI7o42ljuQsjFQAWjZCQBk5Ga3Fmf7ZIfs0ufLTjK8ct71L58n/PrL+af/ABVAHK32ktbfEXwl/Z9i6afY6fewlooj5cAIgCKSOFyFOB3wfSr2q+BdE1rVk1K/Opm6jYtE0OsXcIiJXaSipKFTI4O0DOTnrW558n/PrL+af/FUefJ/z6y/mn/xVH9fjcNtEee+LfAFhpPgnUl8NW2qy3E80TtB/aF1dCRmuI3d/LeRgW4JLYz15612+rz61CsX9g6fYXrEnzBe3z2wUdsbYZM9/T8at+fJ/wA+sv5p/wDFUefJ/wA+sv5p/wDFUdLAYP27xv8A9C94f/8AB9P/APIdaWkT63MJf7e0/T7IjHlfYr97nd1zndDHt7dM59quefJ/z6y/mn/xVHnyf8+sv5p/8VQBNRUPnyf8+sv5p/8AFUefJ/z6y/mn/wAVQAWv+pb/AK6Sf+hmuRtfF1zpvibxBaeILbWHt47tP7Oa20S5nQwmFCcPDEwPzl+pJrqbaZxEcW0p/eP0K/3j71L58n/PrL+af/FUdQOd0afSNW8Y3Os2SaxHevYpaut7pdxaxeWkjMCpliXLZkPAJ47cE1h+OpNUsPiB4f1LTBLFHFY3cUlyNFudSRCzQkKUgIYE7Tgk44Nd958n/PrL+af/ABVHnyf8+sv5p/8AFUdrdL/jf/MO/n/X6Hll3qOva74n8LRzSy30Ntq6zyiPwlf6eIlEMq72lmdkxlgMYzyOeK9ZqHz5P+fWX80/+Ko8+T/n1l/NP/iqfS3z/L/IOt/66/5nN+BrO5tJPE32u3lg87XriWLzUK+YhVMMueoODyOOK6iRBLG0bFgGBBKsVPPoRyPqKj8+T/n1l/NP/iqPPk/59ZfzT/4ql0S7JL7lYOrfm2YP/CB6R/z+eIP/AAo9Q/8Aj9Q+Aba/s9N1W21FtQZYtWuVtTfyyyuYAw2YeQlmXHQ5NdJ58n/PrL+af/FUefJ/z6y/mn/xVC0f9eX+QPVBef8AHjP/ANc2/lVPXNA07xHZR2urRyvHDMs8ZhuJIHSRfusHjZWBGexqe6mc2cwNtKAY25JXjj61L58n/PrL+af/ABVAGPaeD9NsLpLq3udaeSPJVZ9cvZkPHdHlKt9CDWV4F0W4k+Duk6PqH23TZ3sBDLszDPDnIOMjKt+GRXW+fJ/z6y/mn/xVHnyf8+sv5p/8VR0a7/8AB/zDZ3Rkf8IdpEX9imxheybRBsszbttxGV2tE2c7kYAZB5yAc5Ga474d+HtW0PxHYxarYywGHw+IJHxuQSfapG2bxlSdpBwD0NekefJ/z6y/mn/xVHnyf8+sv5p/8VTu73/rZr9Q6W9PwdzBfwFobXt1dR/2pbyXczTzra6zeQRySN1YokoXJwO1cle+GdUtPEvnLYzvbTeK7O5idSZT5CWgjMjEEkAMpBLY/WvS/Pk/59ZfzT/4qjz5P+fWX80/+KpLRprpb8Gn+gPVNd7/AIpr9SauV8DWdzaSeJvtdvLB52vXEsXmoV8xCqYZc9QcHkccV0nnyf8APrL+af8AxVHnyf8APrL+af8AxVC0d/K34p/oD1Vv66/5kkiCWNo2LAMCCVYqefQjkfUVzv8Awgekf8/niD/wo9Q/+P1vefJ/z6y/mn/xVHnyf8+sv5p/8VQBzfgG2v7PTdVttRbUGWLVrlbU38ssrmAMNmHkJZlx0OTXSXn/AB4z/wDXNv5UefJ/z6y/mn/xVRXUzmzmBtpQDG3JK8cfWjovkHVluiofPk/59ZfzT/4qigCHU5dSis92jWlrd3O4fu7u6a3THc7ljc59tv41g6L4o1a98Y3fh/VdJ0+3ktLJLqWax1J7kRl2IRGDQx4JCs3fge9aHiq61Kz0SSfS7uwsEjDPc318GZbaIKSzqgxvYY6Egd+eh5jw3qKaDpcs1j4T8T3kd2/2i61a4jgM12xAHmtGZRL0AwojGBgBR0pd3/X9f13sPt/X9f539dvwx4mudW+HMHiK+hi+0NbSTPFDlUO0twMkkfd961tA1X+3PDem6t5PkfbrWO48rdu2b1Dbc4GcZ64Fct8OPC2kWfw4t47OKwMmpWzC6vrCII1wGLYLMVDEgMR8wyDkYrE0WTU7a2m0zw7N44v7HSZm05ZIxowQGH5cL5gVyOnJHNV9pr+vP9A6Jr+u36nV6b4yt5/iBrXhq/vLCCe1aH7DbmULPcBot7naWy2Ofujgda0dUufEsN4F0XSdKu7baCZLvVJLd93cbVt5Bjpzu/Cue8BSeb4n8UPcJqUV9vtFuYtSjtxICIAA26B2Rtw+Y4CgZxirGuyeNtH0LUdTTW9AlWztpbgRHRJgWCKW27vtftjOPwqZPlV2OK5nZE0HiXXoPFmmaNrmh6fbrqMc7xz2WpyXGzygpO5Wgj67gM5rq68/tNC8W67e6D4om8QaLFPBZuY4U0SXbtnVCwJ+1ZJGwYPHfivQPrVtW0e5Kd9UQr/x/S/9c0/m1TVCv/H9L/1zT+bVNUjCiiigAooooAKKKKAIbu7t7Czmu72eO3t4EMkssrBVRQMkknoBWdovijS/EDsmmyXO5UEgFzZTW/mIejp5qLvX3XI5HqK5v4rz3Q0rQLKzhjn+367awPFM+2NwNzgOf7u5FyO4471oabreuWXjSPw94kawuvtlm93aXljA8A/dsqvG8bO/PzqQwbnngUR1/L7lcJaf152NbWPEul6FLDDqE0puLgExW1rbS3MzgdWEcSs20ZGTjAyPWn2niHSb7w//AG5bX8J0wRtI1y7bFRVzu3bsbSCCCDgggg1harZarpXj8eI9P0ubWba404WMtvbSxJLAySF1ZRKyKytuIPzAjaOD285ubrVpNNvbb7HD/wATLx5HEbV5v3IHlpIUZgOQJUw2OpDAZzSjd6df/tkvnvf8Buy1/rZv9LHr2i+KdK193XTZLncqCQC5sprfeh/jTzUXevuuRyPUVWsPHXh3U9ShsrLUC8lwzLbytbypDclfvCKZlEcpGDwjHofSsa68SajouqXmkeNp7GWzm0qe+jv9Pge32pFtEqNGzucgOCGDc8jArlraPW7Tw34Fi8R6fbW+gafe2uy6gl3XJIHl23mxYCxZLJvKPJz2AJIpWb+78W1+nzJd0vv/ACT/AF+R64v/AB/S/wDXNP5tU1Qr/wAf0v8A1zT+bVK6h0ZGGVYYIpDM+28QaTea7daNaX8M+o2cayXFvG24whiQNxHAPHQ898YNZ+o+O/DulXc0F9fOgt3EdxOtrK8FuxxhZJlUxxnkcMw6j1rmPCnh/SvDPxh1XTdBsYbG0TQrVhHEOpM0uWJPLE+pya0/GWrrqNvf+C/Dlut7q99btFcAL+5sY5QQZZm6A4JIT7zccYOaX2U11v8Ag2v0HpzNPbT8k/1Ok1jX9N0GzjudUufKSaQRQrHG0skznoqIgLOcAnCgnAJqXS9VtNZsvtVg0hj3FGWaF4ZEYdQyOAyn2IHBB7159c22rR/FnS9F0WS0UaX4cBhur+JpVi3S7HYRKyFnIjQZ3DAJ9cHq/COvX2rrqdlrUMEepaTeG0uGts+VN8iusigklQVcfKScEHk1WjWnn9ydv69Sdevl+KuP1bxvoGiX0lrqN66SQhWuHjtpZY7YN90yyIpWLPX5yvHPSrOq+J9K0eSGK7mmkmnQyRQWdrLdSsg6vsiVm28j5sY561zHg+KC58I+Kf7QCkz6tqS3m70DsgB9vLCj6YpnhrxRaeHvhb4Ul1JJbjU73TYIrWzto99xdsIwQFX0A5LHCrnkipWsb+n4p/5fMb0dvX8H/XodVH4p0afw8muW16LjT5OI5II2kZ2zt2hFBYtnjaBnIxinaL4j0zxALgaZNIZLVwk8E9vJBLESMjdHIqsMjkEjB7Vk+EtH1bQvCt3JLbWraxqF3PqEloZykMUkrbvL8wKxwBgFgpyckDFZ/haaY/EjWx4hto7PXp7KBo4bZ/MgNojuFKyHDO29m3bkTGVABHJrrb+v+D/TB7X/AK3/AA/pHaWv+pb/AK6Sf+hmpqhtf9S3/XST/wBDNTUgCiiigArn7zxvotlq11prf2lcXVoVFwtlpN1crEWUMAWijZQSpBxnvXQVwmqaZ4j8KXPiPxNot5pt9BcuL6bTrq0dJGWOFUKLOJCAdseRmMjJ9KV7b7DSvojulYMgYZAIzyMH8j0rAsPHXh3U9ShsrLUC8lwzLbytbypDclfvCKZlEcpGDwjHofSrFzrOnz+CZNaupHg06XTzdSP0ZImj3E/XB/OvNbaPW7Tw34Fi8R6fbW+gafe2uy6gl3XJIHl23mxYCxZLJvKPJz2AJIq3v8r7r8b/ANeZN/c5l5/gv69D0HUfHfh3SruaC+vnQW7iO4nW1leC3Y4wskyqY4zyOGYdR61Z1TxPYaRdLBdQapK7IHDWek3V0mD/ALcUbLnjpnNYXjLV11G3v/Bfhy3W91e+t2iuAF/c2McoIMszdAcEkJ95uOMHNaOv358G+AWNmDcXNtBHZ2SN1mnbEUS/ixX9am/u3/p/1/wOhVtbf0ixpniXS/E+k6hNo8s8iWryW8wntJbdkkVclSsiqcjIzxW5WHomiJ4c8Fw6Wj+a8Fu3nTHrNKwLSSH3ZyzH61uVT0diU7q4UUUUhhRRRQAjMqIWchVUZJJwAKw9O8Z6JqmqR6dbXFwlzMjPAtzZTW4uFXqY2kRVkAHPyk8c9K3a8/1y51D/AITjw1J4ssLWz06O/ZLCaxuDcF7p43WMSlkQxgqW4VXBbGWAHIt7A9mzotR8a6FpV81rqFzPCUdY5J/scxt4mbGFecIY0PI4Zh1Fb1cP8S21mTwvqdsum20uhPbH7fcR3Ja7WDB83y4WVULBc4Jk99pxg9dYyWt7pFtLZt5lpPArRNz80ZUYPr0NG6B7kFt4g0m81260a0v4Z9Rs41kuLeNtxhDEgbiOAeOh574waz9S8deHdIvZra+vnVrdlW5ljtZZIbYt0EsqKUi6g/Ow4IPSuY8KeH9K8M/GHVdN0GxhsbRNCtWEcQ6kzS5Yk8sT6nJpJ7bWfDOh+KdNk0Vr+zvpru8j1I3EKwpHMCzCZWYONmSPlVsgDp2lytFS8n+Dt8ikry5fNfirnbax4i0zQlg/tCaTfcsRBDb28lxLLgZJWONWZgBySBgd6LXWLDXfD73+k3K3NtJG4V1BGCMgqQeVIIIIIBB4NcD8OPtD+JNK/tPf9pi8G2HliTr8zP5h+uVTP0FbXhECPUPHcUAAt11h2THTe1tEZP8Ax4n8Sa0cbSa9fwlYzTuk/T8Vc7eiiipKKesaVa67ot5pWooXtbyFoZVU4JVhg4PY1hJoni+K3FpF4sszbqAgnk0fddbemd4mEe738rHtXR+RJ/z9S/kn/wATR5En/P1L+Sf/ABNAFXQdFtfDuhWulWG8wWybQ0jZZySSzE+pJJPbmuesPC/ibRbjUv7G8QaSltfX816I7vR5JXjMhyV3LcoCB/uiur8iT/n6l/JP/iaPIk/5+pfyT/4mjrcOljD8NeHL/SNW1nU9X1K2vrrVXiZvstm1ukYjTYAA0khOR71qa3pv9s+H9Q0zzfJ+220lv5m3ds3qVzjIzjPTNWPIk/5+pfyT/wCJo8iT/n6l/JP/AImk0pKzGm07oj0uy/s3SLOx8zzfssCQ79uN21QM47dKtVD5En/P1L+Sf/E0eRJ/z9S/kn/xNU227slJJWQL/wAf0v8A1zT+bVNVRYX+2SD7TLny05wvPLe1S+RJ/wA/Uv5J/wDE0hk1FQ+RJ/z9S/kn/wATR5En/P1L+Sf/ABNAE1FQ+RJ/z9S/kn/xNHkSf8/Uv5J/8TQBNRUPkSf8/Uv5J/8AE0eRJ/z9S/kn/wATQBV1zQ7LxDpT2Goq5jLLIkkbbZIZFO5JEYdGUgEH+lYc3g65gs9TurLWLu+1+6s2tLfUdRdVNup7KIURVwfmyFySBk8DHTeRJ/z9S/kn/wATR5En/P1L+Sf/ABNAGXe+HZrm2tYrXxBq2ntbwiF5LeWN2nUAcv5qP83H3hhuetRy+DNGk8LJoAgkjs42EsbpK3mxyh94mEhy3mb/AJtxySc5zk1seRJ/z9S/kn/xNHkSf8/Uv5J/8TRuC0VjDtPBdkk13PrF5ea7c3lqbOSbUDHkQH70arGiIAepIXJ4yTgVUg+Htov2GC+1jV9R07T5EltdOupozDGyHKZKoJJNpAIEjsOB1xXT+RJ/z9S/kn/xNHkSf8/Uv5J/8TRsAL/x/S/9c0/m1TVUWF/tkg+0y58tOcLzy3tUvkSf8/Uv5J/8TQBRi8P2sPiy58QrJMbu5tI7N0LDywiMzAgYznLnv6cVi2vgKTTbvUZtI8V63YLqN5JezRRR2br5jnnBkt2bAAAAJOAAK6jyJP8An6l/JP8A4mjyJP8An6l/JP8A4mjb+vn+Yf1+hj6l4Ui1GWxu11O+tNVsYjCmp23lCZ0bG9XVozGwYgHGzAIyMVVi8Iyaf/ZsGk3s8cC6gb/U7iSdvPvX2EAMVwCC2zI4XaoAGOB0XkSf8/Uv5J/8TR5En/P1L+Sf/E0f1+v5h/X6HO6h4Dsr+9v5E1LUrO01Qg6hYWsqLDdHaFJJKF0LKACY2TOOeeaNQ8Dw3PiOHWtO1nUdHuoLEWCLZJbsiwht2AssT7ecdMcKPSui8iT/AJ+pfyT/AOJo8iT/AJ+pfyT/AOJoWmwGX/YF22km0l8S6vJcCXzEvsW6SpxjbtWIRsvXhkbr7DBovhi30jULnUZry71PU7lFilvr5kMnlqSVjUIqoigknCqMk5OTWp5En/P1L+Sf/E0eRJ/z9S/kn/xNHW4eQWv+pb/rpJ/6GamqpbQuYji5lH7x+gX+8fapfIk/5+pfyT/4mgCaiofIk/5+pfyT/wCJo8iT/n6l/JP/AImgCauc1fwcNbkuIr7XtY/s26P7/TI5YlhkUjBTf5fmhT3USAdR0OK3fIk/5+pfyT/4mjyJP+fqX8k/+JoAyG8PyXetXx1B1fRptOSxi05XbyyCW8xmT7uSCqjHIAPPOKz4Ph7aL9hgvtY1fUdO0+RJbXTrqaMwxshymSqCSTaQCBI7DgdcV0/kSf8AP1L+Sf8AxNHkSf8AP1L+Sf8AxNHn/XcDl7XwFJpt3qM2keK9bsF1G8kvZooo7N18xzzgyW7NgAAAEnAAFbGo+HoNWt9Li1C6uZjp11FdByUBnkjB2mQBQMZO7CheQMY6VoeRJ/z9S/kn/wATR5En/P1L+Sf/ABNC0t5W/DYN7+f6hef8eM//AFzb+VTVUuoXFnMTcykCNuCF54+lS+RJ/wA/Uv5J/wDE0ATUVD5En/P1L+Sf/E0eRJ/z9S/kn/xNAE1FQ+RJ/wA/Uv5J/wDE0eRJ/wA/Uv5J/wDE0APnhFxbyQszqsilS0blWGRjII5B9xXPW/g2M6paX2sa1qettYv5lpFfGERwSYI3hYo03MASAX3EZ4wea3vIk/5+pfyT/wCJo8iT/n6l/JP/AImjrcOljB1TwaNYa4hvdf1g6ZdE+fpiyxCKRT95N/l+aFP90SAY46cVctNKuoPE014bgR6bHZRWtnZROwRCGYu5T7oP3FGOgU+uK0vIk/5+pfyT/wCJo8iT/n6l/JP/AImhabA9SjF4ftYfFlz4hWSY3dzaR2boWHlhEZmBAxnOXPf04qjqng+LWbiZdQ1fVJdMnYNNpRlT7PJ7E7PM2nHKb9p6YwcVueRJ/wA/Uv5J/wDE0eRJ/wA/Uv5J/wDE0f1+oGVrXhiHVr611C3v7zStQtEaKO7sTHvMbYLRssiOjLlQeVOCOMc1LYaJaaBoE9nY+YwbzJpZZn3yTSPks7t3Yk/QdAAABWh5En/P1L+Sf/E1FdQuLOYm5lIEbcELzx9KNgLdFQ+RJ/z9S/kn/wATRQBX1iwtNR054tQnuLe3T948lvey2rKB3MkbKwHrziuM8C2/23xPqWs6Tc6q3hxbdbWy+26nc3K3km7dJOgmdsKPlRWHXDEcEVu+PbS0m8H3t5eaV/a506F7uGxZm2TSIpIDIDhx7EH1xnFZdraajP4dTWR8RXERiE3mxW1mLBRjPQpv2D/rrnHep6N/1qN62Xf9P6Qz4P3s9x8O9Pgl026tY4I2EdxK0RS4zI/KBXLDH+0q9eM1zs6zaKzf8JXonjS5ub3VpYLe4sfERSGXzJnMCqi3qbBs2jlVAxzXX+B7PT9Z+Ful217o1stlcW+WsZkMsTjcTuAkySrH5xnPBFcno/hTTPFi3l1o3hzwLpdvbXctr9luvDy3dwhRiuZdskYjY4ztweCOaqXxiv7v9dmd14Tt/s9pcL/ZWuaaC4O3WdS+2O/HVW8+baPbI+lWNU8L2GsXgubu41WOTaFxaavdWyYH+xFIq5564zXM+AdN8Py6jqbW/hfw9aajo12bQ6jpNnGkcpMasSjBdykB9rJk4ORk07xl4B8Hw+C9duofCehx3CafcSLKmmwh1YRsQwO3IOec0pysubyv+A4K75RH0uPQfid4btdOv9X8m6tr154brVbm6jk2CMLlZZGAILk5Arva4rw38P8AwbL4Y0meXwloTzPZwu0jabCWZigJJO3rnvXa9OlXJcr5X0/zIi+ZX7kK/wDH9L/1zT+bVNUK/wDH9L/1zT+bVNUlBRRRQAUUUUAFFFFAGB4s1+fRLawg0+KGXUNUvEsrUTsRGjMrMXbHJCqjHAxngZGc1W0fW9Wg8Xy+G/Ej2NxcPZ/bbS6sYXhWSMMEdWjZ3KspK8hiCG7Yq14xttHudBDeIbx9Pt4J0ljv0k8s2koPyyb8EJjplvl5weDiuQ8Px2snjq88WrrV3q2laZpMkEmsXJRknLOJGEXlIqFI1j5KLyXIySDhR318/utp+P5hLbTy++/+R0uq65qtz4vHhvw69jbTxWQvbm7vYXnVFZ2REWNXQkkqxJLDAA4OeMi58d6rH4H1C5WGyTXtP1SPSpk2tJbmVpo0Dgbg20pIGALAjoTxmrPia48NfbdO1u78TzeGryW0It71XSE3ELYbyys6MjnJBC7d4zxjJrL8MaToum+D9ZvfGMMY0nWNVa7Ca5GGwmERJJhIMKW8vf8AMBt3AcdA4p7P+ve2+7T1Bu+q/rT/ADOzsP7Y06C6uPE2q6bcwRx7w1pp723lgAlixaaTIx9MY71l+HtT8Ua9Faa0RplrpF4RJDYyQyG58hh8rtLv2hiMNs8s4zjd3rkrTS7LVrvxZYfDkwroN7obwf6GcWRvm3qDDj5M7MbynH3c81mafB4dbSvC6eFo0i8bQXNqt4qf8fsQUqLlbr+IR7Q4+f5fu7f4acdXr5fi2v0XydxS0X3/AJL/AD+9Hsq/8f0v/XNP5tU1Qr/x/S/9c0/m1TUhhXCaj4r8RSyeJLzQ00uKx8OSGOWG8SRpbxkiWV8OrAQjDgAlXyeeBXd14vqaeGtQufFh8d6lNpWszXTww2cdy0Jngj4t2SBcLd5HPzLJyduABipbevo/0+/0KSXXv/n93qeivca/q2n2eqaJqemabZ3Fok5h1DTJJ5E3Lu5ZbiMDAI4x2PNM8BavrOveG/7U1xrN0uZnaxe0tnh8y3BwkjKzvgvjcBn7pXvWHrmo6vd/CrRLHU0az1nxD9l0+4ULsaIygeeQP4WEYkOOx+ldlNeadoFrp9rIRbRSyx2VpGkZILY+VAFBwMKeegA5rRq0pW72/r71YzV3FX7X/r8b+hy+ueOtU0/xzpOh2vh+dLG6v1tJ9TuyFRyYmkxCoOW6cscAEEYPa/4j1jWo/FujaDoE9hbPfW9zcTT3to9wEWIxgAKssfUydcnp0qj8QP8AkYPA/wD2Hl/9J5qveLz4E863HjxND37GNu+rxxYxxuCNIMZ6cA5qF8Ovd/kv1Zb+LTt+rDW9b1fw/pOm200ljf61qd6tlbyJA9vArMGbeyF3bCqpOA3JwMrnhdH1vVoPF8vhvxI9jcXD2f220urGF4VkjDBHVo2dyrKSvIYghu2K5nRbLTpPhzN/wkV5eafo0GqyTaNezyPHLZwB/wBw+9xlFHIUvxsKg8HFT+E7BNU+JEviGx1a81uytNNazGqXLIUupJJFYrF5aLGUQRjLKMEuRkkHDj8Wv9aaLyd9/WxL20/rXV+lj0G1/wBS3/XST/0M1NUNr/qW/wCukn/oZqagYUUUUAFcj4u1XxP4e03UNdtX0mXTdPTzmsXgkM00ajLYm3hVYjOB5bDpzzx11eZfEK98L3kN8YtcmHia0GbLTEu5JHe5TJixYsSj5bHzeX0+YMMBgm7P+vy6jSuekrLuthNscZTdsI+bpnGPWuM8KeONU8ReNNQ0u+0CbRrSCxiurdLwj7TIHdl3OoJCfd+7yeMk84HR6brAlmtdN1ErFrJsUu7m2jViseflPzdPvZAGcnFc5Y/8l21j/sA2v/o6aqt76XTX8E/1RN/cb66fmv8AMTXPHWqaf450nQ7Xw/OljdX62k+p3ZCo5MTSYhUHLdOWOACCMHt0+v6zFoGiz38sbTMmFhgj+/PKx2pGvuzEAfWuZ+IH/IweB/8AsPL/AOk81S61pniq58bwalbWWkX+mWMX+g29zqUtuyTsCHlcLBICQpKrzwCx6txK1jbzf5J/8MU/i+X6tFrwdrt94l+HNtq2rxW8V7PFMJktgRGCruny5JOPl9a6ivP/AIUyXz/CmL7fbW8Kr9p8ow3DSFx5smSwKLtOcjAzkYPGcD0Cqe5K2CiiikMKKKKAEbcEOwAtjgE4BNci+seJNG8S6Paa42mXtprEz26ixtpIpLWQRtICS0jCRcIQThCODjtXWTzJb28k8u7ZGpdtqljgDJwBkn6DmvMDdaJJ470S88A6uda1Ca5MWox/b2v0gtGDNIxZ2c2+GC4VSm44G044F8SX9f159Afwt/1/X5nTPrOv67rmo2nhaTTbWz0uT7PPdX9vJP59xtDNGio6bVUMAXJPJI28V1AkdLUSXAVHVN0gRiyg45wcAkfgK8aGleEdNuPFtn49ihOrT6lcXWnC4+aeSKX54zZ/xbtxIPl/NuXntXofhfU3j8P6ZpHiSVpNcTSIrm+ieNmOMbWLHGCSwIx1JB4pbw03svvabf3W17Ddub5v87L7yn4J1LxX4i0PStd1LUNHjs76ETtZwaZKsiqw4AlNwRnpzs/AVM+s6/ruuajaeFpNNtbPS5Ps891f28k/n3G0M0aKjptVQwBck8kjbxXLFPB7+KvDzfC9dNj1IXy/bhoaoqCz2t5ouRH8oH3dofnfjHessaV4R0248W2fj2KE6tPqVxdacLj5p5IpfnjNn/Fu3Eg+X825ee1NtXb6a/p9+/3ryDy9P1+7b8T1DUv+Elnkgi0htLsl8rdPcXUclyA/9xI1aMkdTuLDt8vpT8Na/deIPDN++pW8VvfWVxcWVysDFo2eMkFkJ52kYODyM4964668a6jpen+HvCusa5YaBrE2kxXWp6rqkyKYh9zbGrkB5iwbk/Ku0kg5ArsvDK+HofBrW3hLULbUbGBZFa4guluDJKRudndScuS24/WqtZv+uv8ASJvov66f0zpKKKKkZD9stv8An4i/77FYj+FvBkmo/wBoSaFoT3u/f9pazhMm7+9uxnPvWrqeqQaRZ/abqO6kj3BcWlnLcvk/7ESs2PfGKz9K8Y6RrGrNplob6O9WA3BhvNNubUmMMF3DzY1BGSBxR1Doa32y1HS4h/77FY+p+HPCOt3X2rWdG0XULjGPOu7WKV8emWBNXPD+uW3iTQLXV7FJY7e6UsizABxgkcgEjt61NperWWtWRu9Mm86ASyQl9jL88blHGCAeGUj8OKOoBanTbG1jtrI2ttBGNqRQ7UVR6ADgU+eWwubeSC5kt5oZVKSRyFWV1IwQQeCCO1Ja6naXt5e2lrLvnsJFiuF2kbGZFcDJGD8rKeM9ao6p4osNIvBbXdvqskm0Nm00i6uUwf8AbijZc8dM5oeu4LyL8U9lBCkUMtvHHGoVERlAUDgADsKf9stv+fiL/vsVjWPjXRtQ1eDS4v7RgvLlXaGO90q6tRIEGWw0sag4BHet+nqBUW6t/tkjefFgxoAd49WqX7Zbf8/EX/fYoX/j+l/65p/NqmpAQ/bLb/n4i/77FH2y2/5+Iv8AvsVNRQBD9stv+fiL/vsUfbLb/n4i/wC+xU1FAEP2y2/5+Iv++xR9stv+fiL/AL7FTUUAQ/bLb/n4i/77FVtRi0vVtNnsNReGa1uEKSxmXAdT1BwQcGr9FAEIu7UAAXEIA6DeKPtlt/z8Rf8AfYqaigCH7Zbf8/EX/fYo+2W3/PxF/wB9ipqKAKi3Vv8AbJG8+LBjQA7x6tUv2y2/5+Iv++xQv/H9L/1zT+bVNQBD9stv+fiL/vsUfbLb/n4i/wC+xU1FAEP2y2/5+Iv++xVW5h0u8vLO6uXhkmsnaS3Yy/6tmUqTjOCdrEc9MmtCigCH7Zbf8/EX/fYo+2W3/PxF/wB9ipqKAIftlt/z8Rf99ij7Zbf8/EX/AH2KmooAqW11brEQ08QPmOeXH941L9stv+fiL/vsUWv+pb/rpJ/6GamoAh+2W3/PxF/32KPtlt/z8Rf99ipqKAIftlt/z8Rf99ij7Zbf8/EX/fYqaigDPih0uDU7nUInhF3dIkcsplyWVM7RyeANzdPU1a+2W3/PxF/32KmooAh+2W3/AD8Rf99ij7Zbf8/EX/fYqaigCpdXVu1nMqzxEmNgAHHPFS/bLb/n4i/77FF5/wAeM/8A1zb+VTUAQ/bLb/n4i/77FH2y2/5+Iv8AvsVNRQBD9stv+fiL/vsUfbLb/n4i/wC+xU1FAEP2y2/5+Iv++xR9stv+fiL/AL7FTUUAQ/bLb/n4i/77FVYodLg1O51CJ4Rd3SJHLKZcllTO0cngDc3T1NaFFAEP2y2/5+Iv++xR9stv+fiL/vsVNRQBD9stv+fiL/vsVFdXVu1nMqzxEmNgAHHPFW6hvP8Ajxn/AOubfyoAPtlt/wA/EX/fYoqaigDM13WJNHs0e10y91S5mfy4be0jzlsE5ZzhUXj7zED6niuR8N6rodjr+oX3iTxbocvibUSkElnFqEeLREzst41LbjgliTgFmJOBwK67xLplxrXhbU9Msrs2dxeWskMdwuf3bMpAPHPeuct7jUrXQE0P/hXTmNYhCYYrm0+wtxjqXD7P+2Wfal3/AK/r+vO77f1/X/A+6D4b2t34Y+G9tc63catIBGSbB7Le1t+8bhI4ovNbOQfm3HvwM1wZ0nw/pTSfa4NKn+2X0rxzav8ADa/mnkeaRnEZkLLvI3bRxnAHFev+DtGufD3g/TdKvp1uLi1hCOyklRySFUnnaoO0Z7AdKoXnw60DUJEe9fWZjHMJ4w+vXxEcgOQyjzsKR2xjHanL4riTvExfhzFp2iXWoWltFOkmp3IuBHbeFbzS7WHbEq4AkUqudmclhkmtnUPGd/pllc3l34K19ba1jaWWUS2JARQSWx9pyeBnpmneE9G1DRdZ8QRXEt9JpklzE2nG8v5Lpgnkrvw0jswHmbuD+HFafimyn1HwhrFlZR+ZcXNjNFEm4Dc7IQBk8Dk96U2+W67foOCV7M46fVdb1rxZ4c8QWfgvWTp1ta3Gd89iHbzhHsZR9o6YVs5weehr0aqGg20tl4c021uU2TQWkUci5B2sEAIyOOoq/VyST5VsiIttJshX/j+l/wCuafzapqhX/j+l/wCuafzapqkoKKKKACiiigArgfjBpOoX/gPVbm2128060s9OuJZrW0CqbpgmVDSfeCjByoxuBwTXfVjeMNHuPEHgrWNIsnjS4vrKW3iaUkIGZSASQCcc+hqKibg7F03aSuMg0TSte8K6Zb65plnqUCW8TrFeW6TKrbAMgMCAcE8+9cx4A8N6I3ijWvFGj6PYafb+Y2maeLO2SFWiibEsmFAyWlDDJ/hjXHWuvksr+Hwi1jYSQrqCWPkwu7ERiUJhSSBnG7HbPtVCC0n8G/DeKz0u2a9utL04RQQxoX86VUwOBgnLdTx1JrabXPKS/q//AANPmZQT5Ixf9W/4P5GL460OTU/F+hXl34UbxLpNlbXQmt1W2cLK5j2EpPIoPCtyOR+NVdYtdP8AE/wlay8DaErWpvYUn0mGCK1aPy7lWniaNyqKw2sCCcH3BzXWXsviiG3tTp1npF7MYQLlbi6ltlWTAyUKxyZXOeDg+5qlo2h6x4e8P3RtHsb/AFi9vZL66ErPBA7yNyqkB2UBQACQ2cZI5qLW07fne/3FXvZr+tPzKfheLwlNdapolp4Qt9AvTCpvdOn0+CMzwNkKxMRZJUJ3Dhjg5Bxmse58FeFr74mabp2leGtIs4dFjGp3ktrYxRM0pJW3jLKoOMh3I/2Fzwa6bRdE1V/FVz4j8RfY4bprRbK3tLKR5UhiDl2LSMql2ZsfwqAFxzkmrHhnQ7nSbrXLzUHikudU1F7ndESdsQVY4lJIHIRBkdAScU1un/W+n+fyE9mv68/8jYX/AI/pf+uafzapqhX/AI/pf+uafzapqQwrzH4m23huK2vGOmvD4in2fZdae0kRbOQkBHN6wCRqpAJUP7BSTg+nVyfiOw8V63YajokMOjRWF/HJbm/eeUyxxOCD/o/l7WYA4z5oBPOB92k730/r59Bq3Um8S6ldeTa+HdKuSNZ1NCvnoP8Aj2hGBLcEdBjOFHd2UdMkUfhDClv8K9IgiyEjM6Lk54E8gHNWl+HOgLcJcg6ol0trDaNPb6vdW7SRxLtQMIpFHHJ6dST3rM8LaBqvgXQNE0SyRru4ub9zfTSTz3EMEGZHJUuw2EjYoAABZicHk1Wl/wCvl/XqSr2Xkv8Ahyp8QvCPhiW0hsLPwzoo1nxBdi1S6/s6EyxhstNPuK53Kgds/wB7FbPiDRvCGlaDYW2s+H31GwsYfItLWPTJb5YlVf7iqwU4GN7Y+taMuh3Nx8QrfXJ3iNnZ6c9tbx5O8SySAyMRjAG1EAIOfvVY1eXxFDOp0Gz0u8iZMMl7dyWzI2eoZY5NwPHGFxjqc8TtH1/4b/g/Mr7X9ev9ehzPhG1sZ/hPCutFtV0wtLKlvHuvCsHmsY4GEe4ylFwpUbhlSOcVH8PU07/hKfETeG4v7N0dRAi6U0P2do5gH3zfZyA0KsNoGVXcUJx0J1dE8P6x4Z8P3AsJLG+1S8v5b+6SZnggZpWJZEYBigHGCVbODkc8P0TQNUbxddeJ/EBtILuWzSxhs7GVpY44lcuWaRlUuxJ/uqAOOc5qr+9/Xb8e34k9P67/AIf0jorX/Ut/10k/9DNTVDa/6lv+ukn/AKGampDCiiigAryj4uXsviDw7q1jp8rLp2jGOS/lTpNcb0KQA99oIdvfYP7wHq9ee+I/hLpd34Tv9N8PPfWk9y3mKsutXvkb2kDuzJ5hUk8n7p5OfejrcOh3t1a297ay2t7BHcW8yFJYZUDo6nggg8EH0rzi58FeFr74mabp2leGtIs4dFjGp3ktrYxRM0pJW3jLKoOMh3I/2Fzwa7HTXvrfxDLpUMEg0eyso9tzctJLLNMzNwJXYlwqqMk5OWHPBFM8M6Hc6Tda5eag8Ulzqmovc7oiTtiCrHEpJA5CIMjoCTihfFf+uy/z+QPa39f10PM/Fc+jLf8Ajd/EVjc6nqsDKdMvreF5VsEMK+WqzrlbRlkDOxYpnIbJBrZ1NtNHiXSE+JV1YXWinRI/s898ymynvM/vXbd+73FdpXd2Lbe9bEHh7xToS6xY6D/ZN3Z6ndzXcdzfzypLbNKcspjVGEoBJx86cYB9a0bLSde8L+G9J0fw6un6lFY2aWzvqE727ZUABwURwR/sYGP71JL3Vfy/Jp/8P1evQJbu3n+a/wAtui0Mf4b38dx4f8R2NkZDpum6lNb6eJd2VgMaSKo3c7R5h25/h29q9ArA0TQToHh68juLgXd9dyTXd5chNgllfrhcnCgAKBk4Cjk1v1T6ei/IXf1f5hRRRSGFFFFACModCpzhhg4OD+YrzCC38PQ/ErQ4PCentod3G87X0stk9l9viCMDH+8VWuG37X3ANgKTuGRn02cSm3kFsyLMVPltIpZQ2OCQCCRn3FcjdaH4i8S6tpD+IYdM06y0m8W+UWN1JcSXEqqyqMtFGI1+Yk/eJ6ZHWhfF/X9fLqD+F/1/Xr0OMv18N3CeKY/GphHipbu4On+ax+1pEObY2h+9gDaf3fG7du716T4P1GfXfAeiajfHM99p0M0xU4yzRgkjHTk1T16x8U6vHeaTbPptpp12pjOorNIbiKNhhgsOzbvxkB/MwOu3jFT2X9oWuqzaLpVpFa6Rp+mxJbTSwuQ0x3BVB3Dciqq5A5yw560o6Rs/L8E7/fp6sJXcr+v4tW/rojj4Lfw9D8StDg8J6e2h3cbztfSy2T2X2+IIwMf7xVa4bftfcA2ApO4ZGXfE228NxW14x014fEU+z7LrT2kiLZyEgI5vWASNVIBKh/YKScHdutD8ReJdW0h/EMOmadZaTeLfKLG6kuJLiVVZVGWijEa/MSfvE9MjrUviOw8V63YajokMOjRWF/HJbm/eeUyxxOCD/o/l7WYA4z5oBPOB92i7su/9den9Pcatfy/r7/6RlfFvSNSvvh5qVxF4gu7K2stNnkuILRVQ3jhMruf7ypwcquNwOCa63T/+RQtf+vFP/RYqjr/hqS++G994Z0yVQ8mmNYwSXLHGfL2KWIBP1IFacNu9n4cS2lKl4bQRsV6EhMHH5VWickvL9SdbRv5/oX6KKKQyHdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAN1z/zxi/7+n/4mjdc/wDPGL/v6f8A4miigA3XP/PGL/v6f/iaN1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAiVrj7ZJ+6iz5aZHmH1b/ZqXdc/wDPGL/v6f8A4miigA3XP/PGL/v6f/iaN1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAN1z/zxi/7+n/4mjdc/wDPGL/v6f8A4miigA3XP/PGL/v6f/iaN1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAN1z/zxi/7+n/4mjdc/wDPGL/v6f8A4miigCJWuPtkn7qLPlpkeYfVv9mpd1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAN1z/zxi/7+n/4mjdc/wDPGL/v6f8A4miigA3XP/PGL/v6f/iaN1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAitmuPKO2KIjzH6yH+8f8AZqXdc/8APGL/AL+n/wCJoooAN1z/AM8Yv+/p/wDiaN1z/wA8Yv8Av6f/AImiigA3XP8Azxi/7+n/AOJo3XP/ADxi/wC/p/8AiaKKADdc/wDPGL/v6f8A4mjdc/8APGL/AL+n/wCJoooAN1z/AM8Yv+/p/wDiaN1z/wA8Yv8Av6f/AImiigCK6a4+xzboogPLbJEhPb/dqXdc/wDPGL/v6f8A4miigA3XP/PGL/v6f/iaN1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAN1z/zxi/7+n/4mjdc/wDPGL/v6f8A4miigA3XP/PGL/v6f/iaN1z/AM8Yv+/p/wDiaKKADdc/88Yv+/p/+Jo3XP8Azxi/7+n/AOJoooAN1z/zxi/7+n/4morprj7HNuiiA8tskSE9v92iigCXdc/88Yv+/p/+JooooA//2Q==)

그림 9 서울시 주택 종류별 주택 통계

1. 서울시 유기동물 보호현황[.CSV]

* 서울시 열린 데이터 광장에서 제공, 구별 유기동물 종류(개, 고양이, 기타)에

따른 관리(처리) 방법(인도, 입양, 안락사, 기증) 데이터

테이블이(가) 표시된 사진

자동 생성된 설명

그림 10 서울시 유기동물 보호현황

7) 서울시 반려동물 유무 및 취득경로[.CSV]

* 서울시 열린 데이터 광장 제공, 구별 유기동물 유무 및 취득 경로(유상입양, 무상입양, 유기동물입양 등) 데이터

테이블이(가) 표시된 사진

자동 생성된 설명

그림 11 서울시 반려동물 유무

8) 유기동물 분석 보고서[.PDF]

* 동물 자유 연대 제공, 지구별 유실•유기동물 처리방법 (자연사, 안락사, 입양, 반환 등)데이터

테이블이(가) 표시된 사진

자동 생성된 설명

그림 12 유기동물 분석 보고서

9) (일반인대상) 입지 선정 요인 중요도 설문조사[.CSV]

* 온라인 설문조사, 입지 선정 시 요인 별(지역주민, 시설 및 인프라, 유기동물 관리방법, 단독/다세대 주택) 상대적 중요도 관련 데이터

|  |  |
| --- | --- |
|  |  |

그림 13 입지 선정 요인 중요도 설문조사

10) 전국 행정구역 (시군구) [.SHP]

* 지오서비스 제공, 시군구 공간 정보

지도이(가) 표시된 사진

자동 생성된 설명

그림 14 시군구 공간정보

## 데이터 정제 방안

* QGIS에서 포인트 벡터의 버퍼를 만드는 데이터 정제 내용
* 자세한 것은 7.1 데이터 정제 방안 참고

# 분석 프로세스

전체 프로세스는 ‘CRISP-DM’을 참고하여 ‘데이터 준비’, ‘데이터 분석 및 모델링’ ‘결과 도출 및 시각화’ 3 단계로 진행

## 분석 프로세스

### 데이터 준비

공공 유기동물 보호소 입지 선택을 위한 변수 선정을 위해 인구통계, 동물 내 특성, 동물 외 특성 변수로 나누어 수집/정제하여 사용 가능한 데이터 셋 준비

* 인구 통계는 사람과 유기동물 발생 현황 간의 관계 분석을 위해 성별, 연령, 소득 수준, 가구 원 수, 주택 유형, 직업 등 사용
* 동물 내 특성은 반려동물 양육 행태를 파악하기 위해 주택유형별 동물 개체 수, 반려동물 유무, 양육인 수, 동물 등록 수 등 사용
* 동물 외 특성은 동물에 영향을 주는 환경 및 인프라 파악을 통한 동물 유실, 유기 현황 파악을 위해 동물 보호소 수, 동물 병원 수, 동물 처리 유형, 개체 별 유기동물 수 등 사용

### 데이터 분석 및 모델링

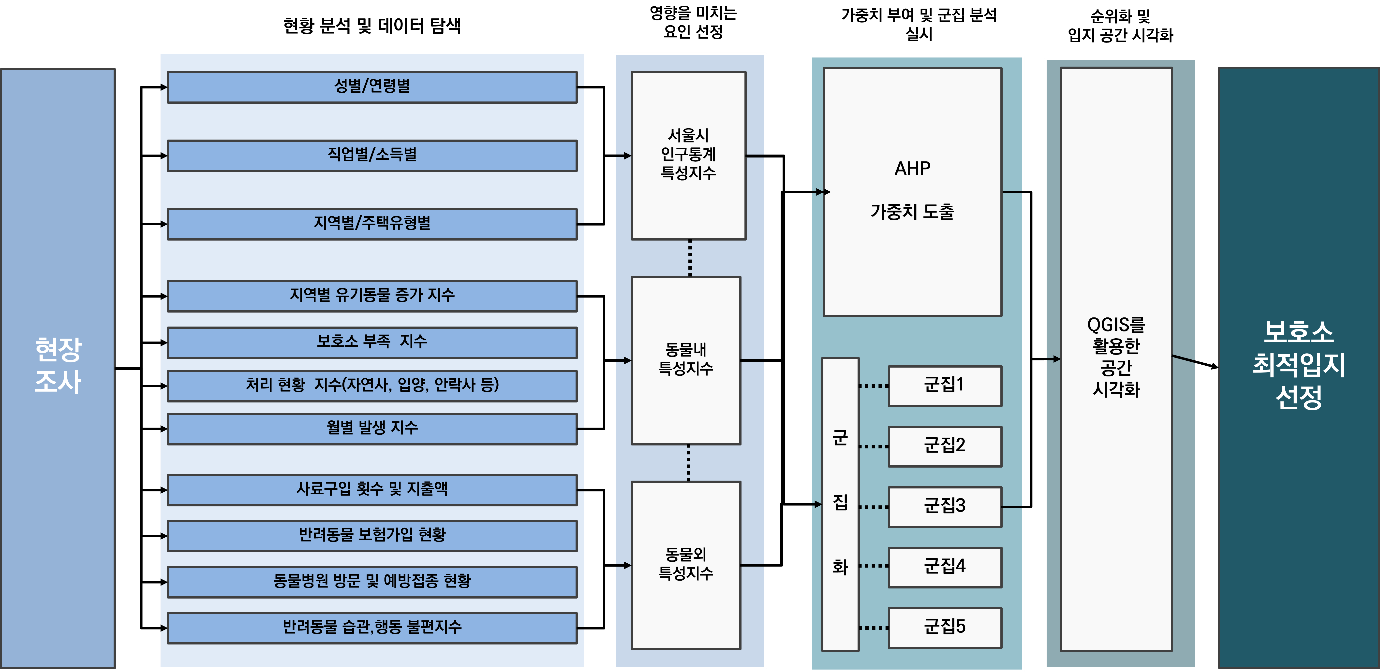


그림 15데이터 분석 프로세스

* 과거 유기동물 현황 조사를 통해 반려동물 유기에 영향을 미치는 요인을 도출하는 탐색적 자료 분석 (EDA) 실시
* 변수 스케일링 (Normalizer)을 통한 데이터 전처리 및 PCA 분석과 FA 분석을 활용한 주요 요인 8개 산출
* AHP분석을 통한 주요 요인의 중요도 산출 및 가중치를 적용한 순위 산정.
* 5가지 군집화 기법(계층적 군집화, 가우시안 혼합 모델. K-Means, DBscan, Mean-Shift) 활용 분석
* 실루엣 계수를 활용한 군집 평가 후 AHP 및 군집화 결과를 반영하여 상위 5개 구 선정
* 동물 보호센터 입지 선정에 필요한 Feature 선정

### 결과 도출 및 시각화

* 군집 평가를 반영한 군집화 결과 시각화 서울시 공공 유기동물 보호소 우선 순위 구 시각화
* 선정한 Feature을 고려한 최종적 최적 위치를 QGIS에서 시각화
* 우선 순위 구와 선정 Feature을 고려한 최우선 보호소 입지 선정

## 분석 내용 및 방법

### EDA

#### 각 변수 간 스케일링 진행

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 16 전체 변수에 대한 Z정규화의 일부 발췌

#### 상관 행렬 시각화 및 변수 제거

* 초기 변수에 대한 상관분석(두 변수 간에 어떤 선형적(또는 비선형적) 관계를 갖고 있는지 분석하는 방법) 실시하고 R의 상관 행렬을 사용하여 시각화
* 변수가 많으므로, ‘인구통계’, ‘동물 내 특성’, ‘동물 외 특성’의 3가지로 나누어 상관 분석 실시
* 상관분석 실시 전 Z 정규화를 통해 스케일링 진행
* 각 부분별로 상관 계수 시각화를 통해, 다른 변수에 비해 상대적으로 색이 연하거나, 흰색에 가까운 변수들은 상관성이 낮다고 판단되어 제거

##### 인구통계

|  |  |
| --- | --- |
| 그림 17 인구통계에 대한 상관 행렬 | 그림 18변수 제거 후의 상관 행렬 |

* 상관 행렬 시각화 후 상대적으로 색이 연한 부분은 상관성이 낮다고 판단하여 제거
* 제거된 변수는 ‘단독주택’, ‘연립주택’, ‘월 소득’, ‘농림어업 숙련 종사자’, ‘100세 이상’, ‘기타’ 임.

##### 동물내 특성

|  |  |
| --- | --- |
|  |  |

* ‘유기된 동물 중 개의 개체수 (개\_유기)’, ‘유기된 동물 중 고양이의 개체 수(고양이\_유기)’, ‘유기된 동물 중 그 외의 개체 수(기타\_유기)’, ‘반려동물의 유무 여부(반려동물 유무)’ 변수 제거
* 각 동물에 대한 유기는 ‘총\_유기동물 수’ 라는 변수로 통합했기 때문에 제외, ‘반려동물 유무’는 상관계수가 낮아 제거
* 전체 상관 행렬 시각화 진행 후 ‘동물 외 특성 (동물병원, 동물보호소)’에서 동물 보호소의 상관계수가 낮아 제거

#### 요인 결정

* 인구통계 데이터를 ‘주거형태’, ‘가구원수’, ‘직업’, ‘나이’ 4가지로 구분
* 동물 내 특성 부분은 ‘반려인의 주거형태’, ‘유기동물 관리(처리)방법 (입양, 자연사, 안락사 등)’, ‘기타’ 3가지 구분

### FA

* 변수들 사이에 공통적인 요인이 있다는 가정하에, 변수들 간의 상호 관련성을 소수의 잠재 요인으로 집약하는 기법 (장원중 & 이정인, 2021)
* 각 변수 값에 대해, Z 정규화 방법으로 변수 스케일링 실시
* R을 통해 Scree plot을 활용하여 시각화, 잠정적인 요인 수 결정
* 스케일링 된 변수를 행렬로 변환하여, FA 실시 후 도출된 MR값과 곱하여 결과값 산출

#### 분석 목적

* 변수들 간의 상호 관련성 파악하여 차원 축소

#### 분석 과정

##### 변수 별 FA

* 각 변수에 대해 FA 실시 및 Scree plot으로 시각화
* 가구원 수

|  |  |
| --- | --- |
| 그림 19 Scree plot | 그림 20 FA 분석 결과 |

* 가구원수 별 Scree plot을 그린 후, 요인의 개수를 2개로 결정
* 결정된 요인으로 FA 분석을 실시했을 때, 2가지 요인으로 묶임
* MR1(1인가구), MR2(2~7인이상 가구)
* 직업

|  |  |
| --- | --- |
| 그림 21 Scree plot | 그림 22 FA 분석 결과 |

* 직업 별 Scree plot을 그린 후, 요인의 개수를 2개로 결정
* 결정된 요인으로 FA 분석을 실시했을 때, 2가지 요인으로 묶임
* MR1 (‘단순 노무 종사자’, ‘판매’, ‘서비스’, ‘장치 및 조립’, ‘기능’), MR2 (‘전문가 및 관련 종사자’, ‘관리자’, ‘사무종사자’)
* 연령

|  |  |
| --- | --- |
| 그림 23 Scree plot | 그림 24 FA 분석 결과 |

* 연령 별 Scree plot을 그린 후, 요인의 개수를 3개로 결정
* 결정된 요인으로 FA 분석을 실시했을 때, 3가지 요인으로 묶임
* MR3 (20~39세), MR2 (60~89세), MR1 (그 외)
* ‘20~24세’의 경우 MR3와 MR2에 대한 FA Loadings가 비슷하기 때문에 비슷한 의미를 지닌 MR3로 묶음
* ‘85~89세’의 경우 MR2와 MR1에 대한 FA Loadings가 비슷하기 때문에 비슷한 의미를 지닌 MR2로 묶음
* 주거형태

|  |  |
| --- | --- |
|  |  |
| 그림 25 Scree plot | 그림 26 FA 분석 결과 |

* 주택 형태 별 Scree plot을 그린 후, 요인의 개수를 1개로 결정
* 결정된 요인으로 FA 분석을 실시했을 때, 1가지 요인으로 묶임
* MR1 (‘아파트’, ‘다세대주택’)
* 반려동물

|  |  |
| --- | --- |
| 그림 27 Scree plot | 그림 28 FA 분석 결과 |

* + 반려동물 별 Scree plot을 그린 후, 요인의 개수를 2개로 결정
  + 결정된 요인으로 FA 분석을 실시했을 때, 2가지 요인으로 묶임
  + MR1 (다세대 주택에서 키우는 반려 동물 개체 수, 연립주택에서 키우는 반려 동물 개체 수), MR2(아파트에서 키우는 반려동물 개체 수, 단독주택에서 키우는 반려동물 개체 수)
    - 반려동물 보호현황

|  |  |
| --- | --- |
| 그림 29 Scree plot | 그림 30 FA 분석 결과 |

* 반려동물 보호 현황 별 Scree plot을 그린 후, 요인의 개수를 2개로 결정
* 결정된 요인으로 FA 분석을 실시했을 때, 2가지 요인으로 묶임
* 기증 변수는 묶이지 않았으므로 제외
* MR1 (‘입양’, ‘반환’, ‘안락사’), MR2 (‘방사’, ‘자연사’, ‘보호중’)
  + - 동물 외 특성

|  |  |
| --- | --- |
| 그림 31 Scree plot | 그림 32 FA 분석 결과 |

* 동물 외 특성의 Scree plot을 그린 후, 요인의 개수를 1개로 결정
* 결정된 요인으로 FA 분석을 실시했을 때, 1가지 요인으로 묶임
* MR1 (동물 등록된 개체 수, 반려동물을 가진 인구, 총 유기 동물 수)

##### 최종 FA

* + - 전체 FA

|  |  |
| --- | --- |
| 그림 33 Scree plot | 그림 34 FA 분석 결과 |

* 전체 변수에 대한 FA로, 3가지 요인으로 묶임
* MR1 (인구특성), MR2 (유기동물 관리방법), MR3(반려인의 주거형태)

- 인구특성 FA

|  |  |
| --- | --- |
| 텍스트, 테이블이(가) 표시된 사진  자동 생성된 설명  그림 35 Scree plot | 그림 36 FA 분석 결과 |

* 인구특성에 대한 FA로, 1가지 요인으로 묶임
  + - 동물 FA

|  |  |
| --- | --- |
| 그림 37 Scree plot | 그림 38 FA 분석 결과 |

* + 동물 내 특성에 대한 FA로, 3가지 요인으로 묶임
    - 전체 평균 FA

|  |  |
| --- | --- |
| 그림 39 Scree plot | 그림 40 FA 분석 결과 |

* 전체 변수 평균에 대한 FA로, 3가지 요인으로 묶임
* MR1 (인구 특성), MR2 (직업, 반려인의 주거형태), MR3(유기동물 관리방법, 가구원수)
  + - 인구특성 평균 FA

|  |  |
| --- | --- |
| 그림 41 Scree plot | 그림 42 FA 분석 결과 |

* 인구특성 변수 평균에 대한 FA로, 1가지 요인으로 묶임
  + - 동물 내 특성 평균

|  |  |
| --- | --- |
| 그림 43 Scree plot | 그림 44 FA 분석 결과 |

* 동물 내 특성에 대한 FA로, 3가지 요인으로 묶임
* MR1(반려인의 주거형태, 동물 외 특성), MR2(유기동물 보호현황(방사, 자연사, 보호중)), MR3(반려인의 주거 형태(다세대, 연립주택))
* 최종 FA의 6가지 결과로 PCA분석 진행

### PCA

* 인구통계, 동물내 특성, 동물 외 특성 데이터를 종합하여 상관분석 실시 후 그 결과를 참고하여 요인 모델을 설정
* FA model에 loading 곱 가중합을 통하여 FA loading 데이터 셋 생성
* loading 곱 가중합을 통해 만들어진 데이터셋은 인구통계, 동물내 특성, 동물 외 특성을 가지고 있음. 이를 이용하여 전체 변수 및 개별 변수에 대하여 각각 PCA 방법을 통해 차원 축소
* FA model에 산술평균을 통하여 평균 데이터 셋 생성
* 산술 평균을 통해 만들어진 데이터셋은 ‘인구통계’, ‘동물 내 특성’, ‘동물 외 특성’을 가지고 있음. 이를 이용하여 전체 변수 및 개별 변수에 대하여 각각 FA 방법을 통해 차원 축소
* 변수 이름은 다음과 같이 명명하고 진행함

텍스트이(가) 표시된 사진

자동 생성된 설명

그림 45 FA 및 PCA 방법을 통한 차원 축소 프로세스

표 3 차원축소 데이터셋 이름

|  |  |
| --- | --- |
| 이름 | 데이터 셋 설명 |
| FA\_A | Loading 가중합을 활용한 전체 데이터에 FA 방법으로 차원 축소 |
| FA\_B | Loading 가중합을 활용한 개별 데이터에 FA 방법으로 차원 축소 |
| FA\_C | 산술 평균을 활용한 전체 데이터에 FA 방법으로 차원 축소 |
| FA\_D | 산술 평균을 활용한 개별 데이터에 FA 방법으로 차원 축소 |
| PCA\_A | Loading 가중합을 활용한 전체 데이터에 PCA 방법으로 차원 축소 |
| PCA\_B | Loading 가중합을 활용한 개별 데이터에 PCA 방법으로 차원 축소 |
| PCA\_C | 산술 평균을 활용한 전체 데이터에 PCA 방법으로 차원 축소 |
| PCA\_D | 산술 평균을 활용한 개별 데이터에 PCA 방법으로 차원 축소 |

#### 분석 목적

* FA 분석 실시 후, 묶인 요인들에 대해, PCA 분석을 진행
* 분석 대상: 데이터 전체, 인구특성, 동물 내 특성, 각각의 평균값으로 총 6개.

#### 분석 과정

* FA로 결정된 요인에 대해 PCA의 Scree Plot을 사용하여, 몇 번째 주성분까지 사용할지 결정

##### 전체

|  |  |
| --- | --- |
| 그림 46 Scree plot | 그림 47 Biplot |

전체 데이터에 대한 Scree plot으로, 4번째 주성분까지 이용.

##### 전체 평균

|  |  |
| --- | --- |
| 그림 48 Scree plot | 그림 49 Biplot |

전체 데이터 평균에 대한 Scree plot으로, 4번째 주성분까지 이용

##### 인구특성

|  |  |
| --- | --- |
| 그림 50 Scree Plot | 그림 51 Biplot |

인구특성에 대한 Scree plot으로, 2번째 주성분까지 이용

##### 인구특성 평균

|  |  |
| --- | --- |
| 그림 52 Scree plot | 그림 53 Biplot |

인구특성 평균에 대한 Scree plot으로, 4번째 주성분까지 이용

##### 동물내 특성

|  |  |
| --- | --- |
| 그림 54 Scree plot | 그림 55 Biplot |

동물내 특성에 대한 Scree plot으로, 3번째 주성분까지 이용

##### 동물내 특성 평균

|  |  |
| --- | --- |
| 그림 56 Scree plot | 그림 57 Biplot |

동물내 특성 평균에 대한 Scree plot으로, 4번째 주성분까지 이용

### AHP 분석

#### 분석 목적

* AHP(Analytic Hierarchy Process)는 평가기준이 다수이며 상호 배반적인 대안들의 체계적인 평가를 지원하는 의사결정 지원 기법의 일부
* 정성적 요소를 포함하는 다 기준 의사결정에 널리 사용됨.
* 요인 분석을 통해 추출 및 가중치 계산된 5개의 요인(지역 주민 특성, 단독 주택에 살고 있는 반려 동물의 수, 다세대 주택에 살고 있는 반려동물의 수, 유기동물 처리 방법, 주변 인프라)에 부여하는 중요도는 개인마다 다르고, 1~2개가 결정적 평가기준으로 작용하는 경우가 많으므로 결정적 평가기준을 지양하기 위해 사용함. (조윤주, 2007)
* 공공 보호소 위치 선정을 위해 5개 요인의 상대적인 비중을 비교하여 결정하기 위해 설문조사를 통해 53개의 응답을 받음.
* 가중치 값과 정규화 된 변수 값을 곱하여 최종 Score를 계산, 이를 토대로 구별 순위 결정

#### 분석 과정

##### 데이터 수집

* 5개의 요인을 기반으로 설문지 작성 및 조사한 설문 응답 50개

##### 유의성 검증

* RI는 1.12 사용 (Thomas L. SaatyLuis, 1982)
* 각 설문지의 응답의 CR값을 구하여 유의성 검증 실시

테이블이(가) 표시된 사진

자동 생성된 설명

그림 58 각 응답의 CR값, 진한색(<0.1)을 사용

* 응답 유의성의 기준이 0.1이므로, 유의성이 0.1 미만인 응답만 사용할 데이터로 선정

##### 데이터 정제

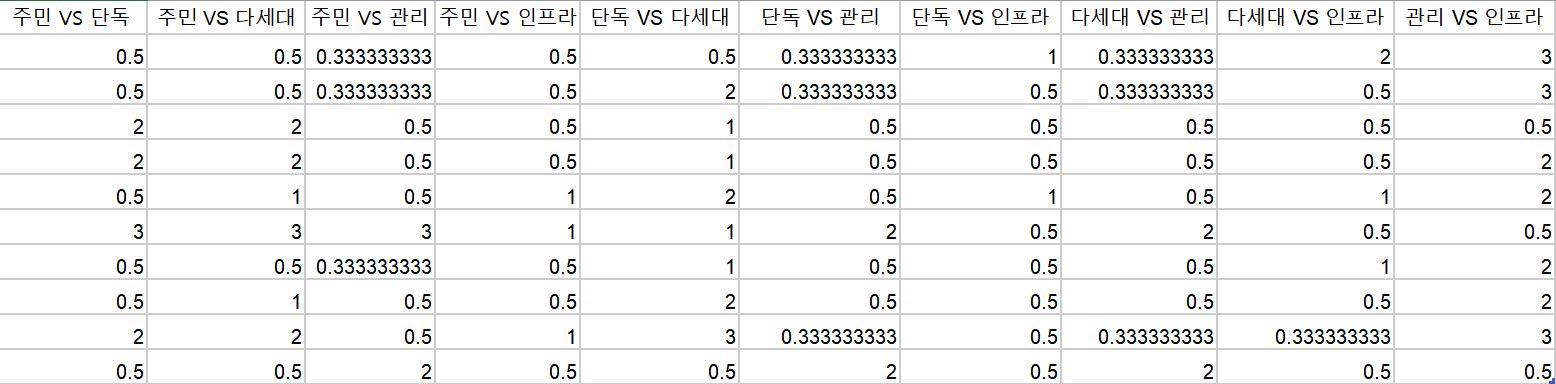


그림 59 데이터 정제

##### 가중치 계산

테이블이(가) 표시된 사진

자동 생성된 설명

그림 60 기하평균을 이용한 가중치 계산

* 속성 간 차이의 평균을 보다 잘 측정할 수 있는 기하평균을 사용하여 가중치 계산

### 군집화

* 최종 분석된 8개의 요인에 대해 군집 분석(계층적 군집분석, 가우시안, K-means, Dbscan, Meanshift) 실시
* 군집의 개수를 설정할 수 있는 분석은 군집의 개수를 4개로 설정
* 군집 평가를 통해 실루엣 계수가 0.4 이상인 군집 기법 활용

#### 계층적 군집분석

* 계층적 군집(Hierarchical Clustering) 분석은 전통적인 군집 방법으로 가장 유사한 개체를 군집해 나가는 과정을 반복하여 원하는 개수의 군집을 형성하는 방법 (우숙영, 이재원, 전명식, 2014)
* FA 분석과 PCA 분석으로 차원 축소된 변수를 sklearn의 StandardScaler() 함수를 사용하여 정규화
* Python을 사용하여 Scipy의 Cluster.hierarchy. dendrogram 활용
* 시각화 후 4개의 군집으로 나누어짐

#### 가우시안 혼합 모델 분석

* 데이터가 K개의 정규분포로부터 생성되었다고 보고 데이터가 모수를 갖는 여러 개의 분포로부터 생성되었다고 가정하는 모델 (이종섭, 2015)
* 군집의 개수는 4개로 결정한 후, 변수를 sklearn의 StandardScaler() 함수를 통해 정규화
* 각 변수별로 군집화 함수 적용(4개의 군)후 결과 저장
* Python을 사용하여 Sklearn.mixture의 GaussianMixture 활용

#### K-Means 군집 분석

* K- 평균 군집(K-Means Clustering) 분석은 주어진 군집의 개수로 각 개체(데이터)를 가까운 초기값에 할당하여 군집을 형성하고, 각 군집의 평균을 계산하여 군집의 중심을 갱신하는 과정을 통해 전체 데이터셋을 상대적으로 유사한 K개의 최종 군집으로 형성하는 방법 (이종섭, 2015)
* Scree plot의 Elbow Effect를 활용하여 데이터 개체 내에서 군집의 개수를 4개로 결정
* R의 kmeans() 함수를 활용하여 군집화 실시

#### DBscan

* 밀도 기반으로 군집을 할당하므로, 군집의 개수는 자동으로 결정
* 어느 군집에도 속하지 않는 노이즈 포인트는 제외하며, 밀도가 낮으면 군집에서 배제 (A Density-Based Algorithm for Discovering Clusters, 1996)
* 각 데이터셋의 속성에 R의 scale() 함수를 통해 표준화

#### Mean-Shift

* 초기값을 임의로 정한 후 확률밀도가 높은 곳으로 군집의 중심점을 이동하며 군을 설정 (곽내정, 권동진, 김영길, 2006)
* 각 데이터셋의 속성에 sklearn의 StandardScaler() 함수를 통해 표준화
* Parameter로 Bandwidth값을 정하고 Estimate\_Bandwidth() 함수를 사용하여 최적화된 Banwidth값 결정

### 공공 유기동물 보호소 최적 입지 선정 모델 개발 및 시각화

* 군집 평가 후 실루엣 계수가 0.4 이상인 Mean-Shift와 K-Means 클러스터링 방법과 AHP의 결과를 활용하여 최종 구별 군집 분석 결과 도출
* 최종 구별 군집 분석 결과로 구별 순위 시각화
* 상위 5개 구 선정 (강북구, 금천구, 서대문구, 종로구, 중구)
* 공공 유기동물 보호소 입지와 관련된 요인 선별: 동물 병원 (유기동물의 감염 및 외상에 대한 신속한 치료 필요), 아파트 • 학교 • 구청 • 소방서 • 공원의 위치 (유기동물이 가장 많이 발견되는 장소), 기존 동물 보호센터의 위치 (동물 보호센터와의 중복 방지)
* 선정한 요인의 위치를 시각화하고(동물 병원, 유기동물 주요 발견 위치, 기존 동물 보호소 위치)반경 500m(도보 10분) 범위로 버퍼를 그린 후, 겹치는 포인트를 Q-GIS로 시각화 하여 최종 입지로 선정

#### 입지 선정 요인 교차영역 시각화 과정

|  |  |
| --- | --- |
| 지도이(가) 표시된 사진  자동 생성된 설명  그림 61 서울시 50m 간격으로 그리드 포인트 생성 | 실루엣, 기어이(가) 표시된 사진  자동 생성된 설명  그림 62 서울시에 위치한 그리드 포인트 추출 |

* 50m 간격으로 그리드 포인트 생성 후, 서울시 그리드 포인트를 선택하여 레이어 생성

|  |  |
| --- | --- |
| 균류이(가) 표시된 사진  자동 생성된 설명  그림 63 서울시 동물병원 위치 500m 버퍼 | 지도이(가) 표시된 사진  자동 생성된 설명  그림 64 그리드 포인트와 버퍼 교차 영역 추출 |

* 각 요인 별 버퍼를 생성한 후, 교차 영역 기능을 통해 서울시 그리드와 교차 포인트 산출

|  |  |
| --- | --- |
| 지도이(가) 표시된 사진  자동 생성된 설명  그림 65 확대한 교차 영역 | 지도이(가) 표시된 사진  자동 생성된 설명  그림 66 교차 영역 포인트 중 버퍼가 겹치는 부분만 선택 |

* 교차영역에서 2번이상 교차된 포인트 추출 및 저장

테이블이(가) 표시된 사진

자동 생성된 설명

그림 67 교차영역에서 2번 이상 교차된 포인트 추출

* 이전 과정을 다른 요인에 동일하게 적용

#### 조인 및 최종 점수 산출

테이블이(가) 표시된 사진

자동 생성된 설명

그림 68 포인트에 교차 값 JOIN 하여 추가

* 교차 값을 서울시 그리드의 ID로 JOIN하여 추가



그림 69 최종 Score 및 ScaleScore 도출

* 교차 값을 스케일링 한 후 합하여 최종 Score를 도출하고, 최종 Score에 Z-Score 변환을 진행하여 ScaleScore 도출

#### 최종 점수에 따른 GIS 시각화

지도이(가) 표시된 사진

자동 생성된 설명

그림 70 최종 점수에 따른 GIS 시각화

* ScaleScore에서 0점(50% 이하) 이하에 해당하는 포인트 제거
* ScaleScore 상위 점수를 갖는 그리드 포인트 추출
* 그리드 포인트를 서울시 공간 시각화하여 위치 파악 후 입지 선정

# 분석결과

## AHP 분석

### 가중치 해석

테이블이(가) 표시된 사진

자동 생성된 설명

그림 71 가중치 해석

* AHP 분석 결과 다세대, 관리가 약 0.27, 0.22의 비율로 가중치가 가장 높음
* 주민, 단독이 약 0.15, 0.14로 비슷한 가중치를 가지는 것으로 확인
* 인프라는 0.06으로 가장 가중치가 낮은 것으로 확인

### AHP Score 계산

|  |  |
| --- | --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명  그림 72 가중치 입력 | 텍스트이(가) 표시된 사진  자동 생성된 설명*그림 73 FA\_B, FA\_D와 가중치 행렬 곱* |

* 변수 FA\_B, FA\_D와 가중치 값 행렬 곱하여 AHP Score 값 2개 산출

|  |  |
| --- | --- |
| 텍스트, 테이블이(가) 표시된 사진  자동 생성된 설명  그림 74 FA\_B score 및 rank | 텍스트, 테이블이(가) 표시된 사진  자동 생성된 설명  그림 75 FA\_D score 및 rank |

* AHP Score에 따른 상위 5개 구 선정
* FA\_B의 경우 종로구, 중구, 금천구, 강북구, 서대문구 순
* FA\_D의 경우 중구, 금천구, 종로구, 서대문구, 성동구 순

## 군집화

### 군집화

사용되는 8개의 데이터셋(FA\_A, FA\_B, FA\_C, FA\_D, PCA\_A, PCA\_B, PCA\_C, PCA\_D)를 3장에 설명한 클러스터링 방법 5개를 활용하여 군집화 진행

### 군집 평가

실루엣 계수를 통한 각 요인 별 군집 평가 및 상위 5개의 방법 및 데이터셋 추출

테이블이(가) 표시된 사진

자동 생성된 설명

그림 76 실루엣 계수를 통한 각 요인 별 군집 평가

텍스트, 테이블이(가) 표시된 사진

자동 생성된 설명

그림 77상위 5개 방법 및 데이터셋

실루엣 계수가 0.4 이상인 조합을 선정하여 PCA을 활용한 차원 축소 진행

|  |  |
| --- | --- |
| 그림 78 Mean-Shift 군집화 결과 | 그림 79 K-Means 군집화 결과 |

## 구별 우선 순위 시각화

### 군집화 결과 시각화

|  |  |
| --- | --- |
| 지도이(가) 표시된 사진  자동 생성된 설명  그림 80 K-means 군집화 결과 | 지도이(가) 표시된 사진  자동 생성된 설명  그림 81 Mean-Shift 군집화 결과 |

### 군집화 결과 구별 순위 시각화

* 색이 진할수록 공공 유기동물 보호소 설립이 시급한 지역
* K-Means를 활용한 상위 5개 구 결정: 중구, 종로구, 용산구, 관악구, 금천구
* Mean-Shift를 활용한 상위 5개 구 결정: 중구, 종로구, 서대문구, 강북구, 금천구

|  |  |
| --- | --- |
|  |  |
| 그림 82 K-means 순위 결과 | 그림 83 Mean-shift 순위 결과 |

#### AHP 분석을 통한 구별 순위 도출 및 시각화

* AHP 분석 후 AHP Score에 따라 순위 계산

|  |  |
| --- | --- |
| 테이블이(가) 표시된 사진  자동 생성된 설명  그림 84 FA\_B 순위 계산 | 테이블이(가) 표시된 사진  자동 생성된 설명  그림 85 FA\_D 순위 계산 |

#### 순위 시각화

* 색이 진할수록 공공 유기동물 보호소 설립이 시급한 지역
* FA\_B 요인을 활용한 AHP 분석 결과 상위 5개 구: 중구, 종로구, 강북구, 서대문구, 금천구
* FA\_D 요인을 활용한 AHP 분석 결과 상위 5개 구: 중구, 종로구, 성동구, 서대문구, 금천구

|  |  |
| --- | --- |
| 그림 86 FA\_B 요인 AHP 분석 결과 | 그림 87 FA\_D 요인 AHP 분석 결과 |

### 군집화 결론

군집 분석 결과 및 AHP 분석 순위의 상위 5개 구 중 겹치는 구를 우선 선정

* 중구, 종로구, 금천구(모든 분석에서 상위 5개 구로 선정됨)
* 서대문구(3개의 분석에서 상위 5개 구로 선정됨)
* 강북구(2개의 분석에서 상위 5개 구로 선정됨)

지도이(가) 표시된 사진

자동 생성된 설명

그림 88 유기동물 보호소 설치가 필요한 상위 5개 구

최종적으로 유기동물 보호소 설치가 필요한 **상위 5개 구(중구, 종로구, 금천구, 서대문구, 강북구)** 선정

## 최종 입지 선정 시각화

|  |  |
| --- | --- |
| 지도이(가) 표시된 사진  자동 생성된 설명  그림 89 최종 입지 그리드 포인트 | 지도이(가) 표시된 사진  자동 생성된 설명  그림 90 상위 5개구의 최종 입지 그리드 포인트(노란색) |

* 최종 입지 그리드 포인트를 상위 5개 구별로 시각화 하여 입지 후보지 선정
* 이때 그리드 포인트와 기존 동물보호센터 버퍼가 겹치지 않게 후보지 선정

### 중구

|  |  |
| --- | --- |
| 그림 91 중구 입지 후보지 시각화 | |
| 그림 92 중구 입지 후보지1 | 그림 93 중구 입지 후보지 2 |

* 기존 동물보호센터(빨간색 버퍼)와 겹치지 않는 입지 후보지 2개 선택
* 그림 92의 경우 남산과 겹쳐 있으므로 동물보호센터 설치가 용이하지 않아, 그림 93로 최종 입지 후보지 선정

### 종로구

|  |  |
| --- | --- |
| 그림 94 종로구 입지 후보지 시각화 | |
| 그림 95 종로구 입지 후보지 1 | 그림 96 종로구 입지 후보지 2 |

* 기존 동물보호센터(빨간색 버퍼)와 겹치지 않는 최종 입지 후보지 2곳 선택
* 그림 95의 경우 창덕궁 내부이므로 동물보호센터 설치가 용이하지 않아,  
  그림 96으로 최종 입지 후보지 선정

### 금천구

|  |  |
| --- | --- |
| 지도이(가) 표시된 사진  자동 생성된 설명  그림 97 금천구 입지 후보지 시각화 | 그림 98 금천구 입지 후보지 1 |
| 그림 99 금천구 입지 후보지 2 | 그림 100 금천구 입지 후보지 3 |

* 기존 동물보호센터(빨간색 버퍼)와 겹치지 않는 최종 입지 후보지 3곳 선택
* 그림 98의 경우 독산자연공원 및 학교 등으로 인해 설치가 용이하지 않음
* 그림 99의 경우 한강 및 도로변이므로 설치가 용이하지 않음
* 따라서 그림 100으로 최종 입지 후보지 선정

### 서대문구

|  |  |
| --- | --- |
| 그림 101 서대문구 입지 후보지 시각화 | 그림 102 서대문구 입지 후보지 1 |
| 그림 103 서대문구 입지 후보지 2 | 그림 104 서대문구 입지 후보지 3 |

* 그림 102의 경우 서대문구의 가장 왼쪽에 위치해 있고 그림 101에서 다른 구의 보호소가 거리상 멀지 않기 때문에 제외(같은 이유로 그림 101의 아래쪽 후보지도 제외)
* 그림 103, 그림 104를 살펴보면 서로 멀지 않은 거리에 있고 기존 보호소, 산, 강과 같은 제한 요소가 없기 때문에 두 후보지 사이 정중앙에 입지를 하는 것이 적절하다고 판단

### 강북구

|  |  |
| --- | --- |
| 그림 105 강북구 입지 후보지 시각화 | |
| 그림 106 강북구 입지 후보지 1 | 그림 107 강북구 입지 후보지 2 |

* 그림 106의 경우 강북구청을 비롯하여 아파트, 학교와 같은 입지 선정 고려 요인이 주변에 존재하며, 특별한 다른 제한 요소가 없음
* 그림 107의 경우도 아파트, 학교, 공원 등의 입지 선정 고려 요인이 주변에 존재하며, 특별한 제한 요소가 없어 보임
* 따라서 현장조사 및 주민의견 수렴을 통해 두 후보지 중 최종 입지 후보지 선정

# 활용 방안

## 문제점 개선 방안

* (예산 효율성) 유기동물 보호소 설치 시 동물 병원 위치를 고려한 후보 군 우선 선정으로 관리, 치료, 입양 등의 접근성을 강화하여 사회적 비용 감소
* (동물 복지) 보호센터 최적 입지를 통해 보호센터의 수용능력 증가에 따른 안락사 감소
* (상생 정책) 유기동물 관련 정책마련을 통해 시민사회에 동물 보호 및 생명 존중 문화 함양

## 업무 활용 방안

### 공공 유기동물 보호센터 설치

* 2020~2024년 동물복지 종합계획 중 “직영 동물보호센터 확충 계획”에 본 프로젝트의 분석 결과를 적용하여 직영 보호 센터 수 증설 가능
* 서울시의 “동물 복지 센터 증설” 정책에도 적용 가능[[4]](#footnote-5) (서울특별시 시민건강국, 2019)
* 직영동물 보호센터 수를 늘리면서 동물 보호 센터에서 등록 대행을 수행하여, 동물 등록 비율 증가 예상[[5]](#footnote-6) (2020년 반려동물 보호·복지 실태조사 결과, 2021)

### 유기동물 의료 정책과 연계

* 프로젝트는 동물 병원의 위치를 고려하여 최적 입지 선정을 진행했기 때문에, 서울시의 ‘동물 공존도시, 서울 기본계획’의 26개 단위 과제 중 ‘유기동물 응급(구조) 치료기관 지정’에 직접적인 도움이 될 수 있음

# 참고자료

* EsterKriegelMartin, SanderJiirg, & XuHans-PeterXiaowei. (1996). A Density-Based Algorithm for Discovering Clusters.
* Thomas L. SaatyG. VargasLuis. (1982). “The Logic of Priorities.”
* 곽내정, 권동진, & 김영길. (2006). Mean Shift와 영역병합을 이용한 칼라 영상 분할. 한국콘텐츠학회.
* 김형석. (2015.11). 예비타당성조사에서의 AHP 분석 및 이해. KDI 공공 투자 관리센터.
* 농림축산검역본부 동물보호과. (2021). “2020년 반려동물 보호·복지 실태조사 결과.” 정부: 농림축산식품부.
* 농림축산식품부. (2020). “2020～2024년  동물복지 종합계획.”
* 서울특별시 시민건강국. (2019). “동물 공존도시, 서울.” 서울특별시: 시민건강국.
* 우숙영, 이재원, & 전명식. (2014). 상대적 계층적 군집 방법을 이용한 마이크로어레이 자료의 군집분석. “한국데이터정보과학회지, 25(5)”, 999-1009.
* 이종섭. (2015). GMM 지원을 위해 k-means 알고리즘을 이용한 어휘 인식 성능 개선.
* 이혜원. (19년 3월). “사설 동물 보호소 실태조사 및 관리 방안 마련 연구 최종보고서.” 농림축산식품부.
* 장원중, & 이정인. (2021). “데이터 분석의 모든 것.” 아이리포.
* 조윤주. (2007). AHP를 이용한 의복평가기준의 우선순위 분석. 한국의류산업학회지.
* 지인배, 김현중, 김원태, & 서강철. (2017). “반려동물 연관산업 발전방안 연구.” 한국농촌경제연구원.
* 황원경, & 손광표. (2021). “2021 한국 반려동물보고서.” KB금융지주: 경영연구소.

# 부록

## 데이터 정제 방안

### 보호센터

현재 운영되고 있는 보호센터와 중복되지 않는 곳을 보호소의 입지로 선정하기 위해, 보호센터 근방 500m는 제외

#### Raw 데이터

* 이름: 서울시 동물 보호센터 위치
* 설명: 서울시 동물 보호센터 위치
* 출처: 동물보호시스템

#### 정제 과정

지오코딩 (공통)

* GeoCoder – Xr을 사용하여 좌표 변환
* GeoCoder – Xr는 지오 서비스에서 제공하는 좌표 변환 프로그램, 도로명/지번 주소를 입력하면 위도와 경도를 SHP파일과 CSV파일로 변환하여 제공

Qgis를 사용하여 동물 보호센터 주변 500m 버퍼 생성

포인트 기준 500m(도보 10분 거리)를 기준으로 버퍼를 생성, ‘보호센터\_버퍼\_500m’ 로 저장

지도이(가) 표시된 사진

자동 생성된 설명

그림 108 동물보호센터 근방 500m 버퍼

### 구청

#### Raw 데이터

* + 이름: 서울시 주민자치센터 정보
  + 설명: 서울시 구청 정보
  + 출처: 서울 열린 데이터 광장

#### 정제과정

1. QGIS를 사용하여 구청 주변 500m 버퍼 생성

포인트 기준 반경 500m(도보 10분 거리)의 버퍼 생성 후 ‘구청\_버퍼\_500m’ 로 저장

지도이(가) 표시된 사진

자동 생성된 설명

그림 109 구청 근방 500m 버퍼

2. 서울시 그리드 파일과 중복되는 부분 추출

* 동물보호센터를 정제하여 만든 ‘서울시 그리드’ 파일과 ‘구청\_버퍼\_500m’ 파일의 교차영역을 사용하여 중복되는 부분을 찾음
* 생성된 교차영역을 ‘구청\_교차영역’로 저장하여 CSV로 내보냄

### 아파트

#### Raw 데이터

* 이름: 서울시 공동주택 가격 및 공간 정보
* 설명: 서울시 아파트의 가격과 위치 정보
* 출처: 국토교통부/공간정보센터

#### 정제과정

1. Qgis를 사용하여 아파트 주변 500m 버퍼 생성

포인트 기준 반경 500m(도보 10분 거리)의 ‘아파트\_버퍼\_500m’ 로 저장

지도이(가) 표시된 사진

자동 생성된 설명

그림 110 아파트 근방 500m 버퍼

2. 서울시 그리드 파일과 중복되는 부분 추출

* ‘서울시 그리드’ 파일과 ‘아파트\_버퍼\_500m’ 파일을 Qgis – 벡터 – 교차영역을 사용하여 중복되는 부분을 찾음
* 생성된 교차영역을 ‘아파트\_교차영역’로 저장하여 CSV로 내보냄

### 소방서

#### Raw 데이터

* 이름: 서울시 소방서 관할 위치정보
* 설명: 서울시 소방서 위치
* 출처: 서울 열린 데이터 광장

#### 정제과정

1. Qgis를 사용하여 소방서 주변 500m 버퍼 생성

포인트 기준 반경 500m(도보 10분 거리)버퍼 생성 후 ‘소방서\_버퍼\_500m’ 로 저장

지도이(가) 표시된 사진

자동 생성된 설명

그림 111 소방서 근방 500m 버퍼

2. 서울시 그리드 파일과 중복되는 구역 추출

* + ‘서울시 그리드’ 파일과 ‘안전센터\_버퍼\_500m’ 파일을 교차영역을 사용하여 중복되는 부분을 찾음
  + 생성된 교차영역을 ‘소방서\_교차영역’으로 저장하여 CSV로 내보냄

### 학교

#### Raw 데이터

* + 이름 : 서울시 학교 기본 정보
  + 설명 : 서울시 학교 위치
  + 출처 : 나이스 교육정보 개방 포털

#### 정제과정

1. Qgis를 사용하여 학교 주변 500m 버퍼 생성

포인트 기준 반경 500m(도보 10분 거리)의 버퍼를 생성하여 ‘학교\_버퍼\_500m’ 로 저장

지도이(가) 표시된 사진

자동 생성된 설명

그림 112 학교 근방 500m 버퍼

2. 서울시 그리드 파일과 중복되는 부분 추출

* + ‘서울시 그리드’ 파일과 ‘학교\_버퍼\_500m’ 파일을 Qgis – 벡터 – 교차영역을 사용하여 중복되는 부분을 찾음
  + 생성된 교차영역을 ‘학교\_교차영역’로 저장하여 CSV로 내보냄

### 공원

#### Raw 데이터

* + 이름 : 전국도시공원정보표준데이터
  + 설명 : 전국 도시공원정보(공원유형, 보유시설 등)을 제공
  + 출처 : 공공 데이터 포털

#### 정제과정

1. 엑셀에서 서울특별시에 해당하는 공원만 추출하여 저장

1. Qgis를 사용하여 공원 주변 500m 버퍼 생성

포인트 기준 반경 500m(도보 10분 거리)의 버퍼를 생성하여 ‘공원\_버퍼\_500m’ 로 저장

지도이(가) 표시된 사진

자동 생성된 설명

그림 113 학교 근방 500m 버퍼

2. 서울시 그리드 파일과 중복되는 부분 추출

* + ‘서울시 그리드’ 파일과 ‘학교\_버퍼\_500m’ 파일을 Qgis – 벡터 – 교차영역을 사용하여 중복되는 부분을 찾음
  + 생성된 교차영역을 ‘학교\_교차영역’로 저장하여 CSV로 내보냄

## 사례조사

### 반려동물 인식 및 양육 현황

* + 「2018년 반려동물에 대한 인식 및 양육 현황 조사 보고서에 따르면, 반려동물 양육 여부를 물어본 결과, 전체 2,000명 중 56.5%가 반려동물을 길러본 경험이 있다고 응답함.

※ ‘현재 기르고 있다’는 응답은 27.9%, ‘과거에 길렀으나 현재 기르고 있지 않다’는 응답은 28.6%, ‘한 번도 길러 본 적이 없다’는 응답은 43.6%임.

* + 「2018년 반려동물에 대한 인식 및 양육 현황 조사 보고서에 따르면, 반려동물 양육인을 대상으로 월 평균 지출하는 반려동물 양육비를 물어본 결과, 5만 원 이하라는 응답이 29.8%로 가장 많고, 그 다음으로 ‘5만 원 초과 10만 원 이하(24.8%)’ 등의 순임.
  + 반려동물 수는 해가 갈수록 증가하고 있는 추세임.

※ 2011년 96,268마리 → 2014년 81,147마리 (소폭 감소) → 2019년 135,791마리 (급증)

|  |  |
| --- | --- |
| 그림 114 반려동물 양육 여부 그래프 | 그림 115 월 평균 반려동물 양육비 |

![지도이(가) 표시된 사진

자동 생성된 설명](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAeAB4AAD/4RDaRXhpZgAATU0AKgAAAAgABAE7AAIAAAAFAAAISodpAAQAAAABAAAIUJydAAEAAAAKAAAQyOocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHVzZXIAAAAFkAMAAgAAABQAABCekAQAAgAAABQAABCykpEAAgAAAAMzOAAAkpIAAgAAAAMzOAAA6hwABwAACAwAAAiSAAAAABzqAAAACAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMjAyMTowODowNiAxNjo0Njo0MQAyMDIxOjA4OjA2IDE2OjQ2OjQxAAAAdQBzAGUAcgAAAP/hCxdodHRwOi8vbnMuYWRvYmUuY29tL3hhcC8xLjAvADw/eHBhY2tldCBiZWdpbj0n77u/JyBpZD0nVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkJz8+DQo8eDp4bXBtZXRhIHhtbG5zOng9ImFkb2JlOm5zOm1ldGEvIj48cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiPjxyZGY6RGVzY3JpcHRpb24gcmRmOmFib3V0PSJ1dWlkOmZhZjViZGQ1LWJhM2QtMTFkYS1hZDMxLWQzM2Q3NTE4MmYxYiIgeG1sbnM6ZGM9Imh0dHA6Ly9wdXJsLm9yZy9kYy9lbGVtZW50cy8xLjEvIi8+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDIxLTA4LTA2VDE2OjQ2OjQxLjM3NTwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iPjxkYzpjcmVhdG9yPjxyZGY6U2VxIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpsaT51c2VyPC9yZGY6bGk+PC9yZGY6U2VxPg0KCQkJPC9kYzpjcmVhdG9yPjwvcmRmOkRlc2NyaXB0aW9uPjwvcmRmOlJERj48L3g6eG1wbWV0YT4NCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgPD94cGFja2V0IGVuZD0ndyc/Pv/bAEMABwUFBgUEBwYFBggHBwgKEQsKCQkKFQ8QDBEYFRoZGBUYFxseJyEbHSUdFxgiLiIlKCkrLCsaIC8zLyoyJyorKv/bAEMBBwgICgkKFAsLFCocGBwqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKv/AABEIAbYC8wMBIgACEQEDEQH/xAAfAAABBQEBAQEBAQAAAAAAAAAAAQIDBAUGBwgJCgv/xAC1EAACAQMDAgQDBQUEBAAAAX0BAgMABBEFEiExQQYTUWEHInEUMoGRoQgjQrHBFVLR8CQzYnKCCQoWFxgZGiUmJygpKjQ1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoOEhYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4eLj5OXm5+jp6vHy8/T19vf4+fr/xAAfAQADAQEBAQEBAQEBAAAAAAAAAQIDBAUGBwgJCgv/xAC1EQACAQIEBAMEBwUEBAABAncAAQIDEQQFITEGEkFRB2FxEyIygQgUQpGhscEJIzNS8BVictEKFiQ04SXxFxgZGiYnKCkqNTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqCg4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2dri4+Tl5ufo6ery8/T19vf4+fr/2gAMAwEAAhEDEQA/APpGiisrWvE2keHfJ/tm8W18/Pl7lJ3YxnoPcUAatFcr/wALL8I/9BmL/vh/8KP+Fl+Ef+gzF/3w/wDhSuh2Z1VFYek+M/D+t3f2bTNThmnxkR8qT9MjmtymIKKKRmVELOQqqMkk9BRsAtFZw1SSX5rWzeSPs7ME3fQUv2+8/wCgf/5GH+Fcn1yj0b+5/wCRt7Gf9NGhRWf9vvP+gf8A+Rh/hR9vvP8AoH/+Rh/hR9co+f3P/IPYz8vvX+ZoUVn/AG+8/wCgf/5GH+FH2+8/6B//AJGH+FH1yj5/c/8AIPYz8vvX+ZoUVn/b7z/oH/8AkYf4Ufb7z/oH/wDkYf4UfXKPn9z/AMg9jPy+9f5mhRWf9vvP+gf/AORh/hR9vvP+gf8A+Rh/hR9co+f3P/IPYz8vvX+ZoUVn/b7z/oH/APkYf4Ufb7z/AKB//kYf4UfXKPn9z/yD2M/L71/maFB6Vn/2hdLy+ntjvtlBP5VatrmK7h8yE5GcEEYKn0IrSniKdR8sXr6NfmTKnKKuyWiiitzMKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKDQAUUUUAFFFFABVMTFrgZPGcVLPOFG1eT3qpQBpUUyJ/MQH86fQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFYWu+J4dCuI4Zbd5TIm4FWxjnFZVa0KMeao7I1pUp1pckFdm7RWFoPieHXbiSKK3eIxruJZs5o/wCEjeLxLd6ZeWJt4Le0N0tyZQd6gkH5QOOnrTp1YVYqcHdP9AqUZ0pcs1Z/5m7RXHS+O5bCxt7/AFbTPIs72F5LUxzbnLAZVGGAAWHTBPpTNQ8Y6pYXlpp9zp1tb3t80Ztz5xkj2sSGzwpyvH1zWmt7GXS52lFcZceMb7Tmmj1GG3LW+pxWcjwo5BR4g+4LknIzjvWjNrep2mtada3FravBfyuqNG7b1UIW3EEewH40dL/13/UDoqK53RNbvfEPhm2vtOksVun/ANcjZdU9sBsg/WpNP1XU/wDhJ5dJ1NLVttqtyJLdWXGWZcEEn+7R1sHS5vUVyi+LLxok1L7FF/ZD3f2UN5h80ZfyxJjGMFscehzntTLzxr9g8NtqNykQk/tJ7NYwCcos5Rmx14QFj2GKV/6+7/MdmddS1gf8JH/xV8WkrH5kE1ktxHNGhbksw5I4Aworfp9BdbBRRRQAUUUUAFeafFFQ3irwqGAILz5B+iV6XXmnxP8A+Rr8Kf78/wDJK48d/utT/C/yN8P/ABY+pl/bLX+3/wCy/sq7/I87fgYxnGKJ7y1g1u2002ql7hGcPgYGKrX+jJd6+l3DqUltdLEFMcZGSuc/lUmq6CdSvoLuK9mtZoEKK0eM81+XRWGvHmlZOLvvpK2j9L9j6Z+0s7Lr+A3WbCT7ZpN1pdszXkN9EQ0KZYLkk9O1ezDoM15Dpmu6m2omXQdFn1WGxfbNMHChmA5Az1Nek+HPENr4l0lb2zDphiksUgw0Tjqpr73h+nXpYPlrK2unozwswlCVW8DVqhrHNkifwyTIje4LDIq/VDV/+PeH/r4j/wDQq9fF/wACfoclH+IiYDAwKKKK5iivf39tpljJeX0nlQRAbmwT1OAABySSQAPesDWNW1X/AIQ/XNSW3+wLDYyyWgY5mDBGO5scL2wOfc9h0dxbw3du8FzEssMg2ujjIYehFUtO0k6cJYEuZJrJgBHbzfN5XqAx5K+xzj6cU1YRzcPiW8vvEtra+bZJH/Z11K62d8J8spi2lvlXbjLY+p9KoWWt3l/4dsIy+sR6kNMjmCwbHEu5TtkYn1YHrzxWzpeoXVtpjW2p6Letc7pEd4IYwpXccYww/hxVq18MWht7Ka2lvrJ4bOO1G2XY5jTJUNjjIyfzq9EIx7bUdTvfE2laXdaisMh0yZrhLeT96sg8jl15GeWxx3NN1O81Gz0/xZ9n1W6uILHTgI5mKho7jEjOFKgdFMf0Nad34ek/4SLTfImu/JFrdJNdGYtIpZoCBuJyMhG6dOelWde0ZY/Amp6Xo9t80lpKkUS9WYg9z1JJ6mldANkudL1eOKeTV7iyZV2mL7UYGyD3XIOfrVXQLwHW9asbTVftaxeWLfz7jzcOY9x75I5FWfFWhxX3hDV4rWxjlvprGVY8INzSFCBz65rUi0u2t4d1nbQW9xswsgjHynHX3paWGV9P1sy3w07U7c2eobS6pnckyjGWRu45GQcEZ6d61qzdN0WCwme5kd7q+kGJLqY5dh6Dsq+wwK0ql2GFVrUbNZuUXhXiRyPfLDP5AVZqvb/8hyb/AK90/wDQmqP+XkPX9GUvhl6fqjRpKXtRXqnMGKSlpKAFoooFABRRQKAEpaBQaADFFFBoAKSlooAMUlLRQAlLRRQAlFLRQAlLRRQAUUCigApKUUCgBKr3t9BYWzT3ThI1HJNTswGa838cao11qYtI5AYYRyFPVq48ZiVhqTn16HXg8M8TVUOnU6zTfE9rqxkS2/dujYCuRlh6ir3myHqxrx+KV4JVkiYq6nII7V6pYXseoWMVxEwbco3Y7N3FcuXY54lOM91+J15jgVhmpQ+F/gWzI7cFjTo5WQ9cj0NR5ozXrHkkjzu3Q7R7U3e/94/nTaKAJY52Q8ncPc077U2egxUFLQBYW7Hdf1qN7hn4HyioqKAFoxRRQBLbybHwehq4KzTV+F90YNAD6XFBooASloNBoAKKKx7vxFBaXrW721ywWaOBpVUbQz4x3yfvDtQBsUlLSUALSUvaigAxSUtFACUuKKKAExXJeL/D1/rF9BLZKhVI9p3NjnJrrqK58Rh4YiHJPY3w9eeHqe0hucj4Q8PX+j3s8l6qBXQKNrZ5zUepWt/qHjK5/wCJTeLZz6Y1j9qzFtDFm+bG/djB9M+1dlRTo0IUaapx2V/xv/mOviJ15upLd2/A811bQ9a8RaDpelHTZLSbS4zI8srL5csqLtRVIJJBOTkgYFWtatLzWXl1a50KZV0zTZBBb3GC0s5Ib5QjE4GwDIIJzXoFFbvV36nOtFY4iG30G6jsYLBLizm+2R3cg+wz/vHC7cFmX3xkntXQWeiGLVJdU1C5N3eMpjiO3akEec7UXtnjJ5J/KtfFFD2Ecd4V8Oxy+D9P+1i9tJ1hwyRXEkBzk9VUjn607wV4eRNJ0/Vr99QOqvbqJnuLqUs2M8MrHtk8Ed66+ijqM54eD7cSqn266+wLc/ahY/J5Yk3buu3djdzjOM1hJay6d4fuNQisdl4dTuImuGt97wwPcuS6qRyMHPQjnPIrvqKVv6/r7h33OL8KW0MXia6+wSG9s47RES78hYgpLsTGNqqpHOeBkZ612lIFCjCgAewpaYuoUUUUAFFFFABXmnxP/wCRr8Kf78/8kr0uvNPif/yNfhT/AH5/5JXHjv8AdKv+F/kb4f8AjR9TmNSj1G18YJf2dg11EbYREhwMZbOf0rphyKwrjWja+LlsbmeOK1a23jfgZfdjrVqddR1nxDBoGjTJbSSQmea4YZKIDjgevNfmiwuIxk6NGMUm46PXbz9PI+k9pClGU2+pp/DDXtL0jw3dWGqXkNndW11I0qTMFOCcgjPWtH4XgzWut6gikW17qcstuxGN6cDIq3pnwx8N2Npsu7NdQnc5kuLn5mY/0rI1TRtQ+HVtLq/hq7aTSYmD3GmznKqpIBKHsea/U4KUYpSPmJNNux6PWZqdxHM8NtCfMlWZGZV52gHJye1SXKy6lpcb2U3leYquP9oEZxnqPwqCzlhgItmh+yy/3D0b3B71xYupJv2VrJ9f0Xn6/czWlFJc+7XQu0UUUiSjrOr2uhaTNqF+zCKID5UGWdicKqjuSSABWLceKdR07QX1LVtHEEs00cNlZJcbpJXc4VWOMKc+mcDNU/ibardaTpH2qaW3sYtWge7micqY48MA24dBuK89utcb4jNrpHjPTE0S7kuNEspbe+1EeeZkt28wxq+STtyHJI9FzWkYpols9E0fxFeza7LouvWEVlfC3F1EYJjJHLHnacEgEEHAI9xWFcfExp7y9ttG09Z/JvLa1gmmlKpOZiy7hgcKCh55yOazfGb3/iHxtaweETFeGDTnjvpIpwuyKWRflDjIDEIcegOazWtLi48QeII7vTH0mDTLazu91rcI7W/kJIybQUw2cEY7U1Fbhc6+bxdrNg2u2+p2FnHdabpLajEYZWdH4fCnIB6x/rWpBr17qmgWd94ehsNSkkUeev2zYsbbQSMhW556GsUeGHvvC+rapBf3upX+saK1vF9r8tcBkYouFAA5etdLfU9H8MWNloWnQfbHjVJGkYLHAwUAuwHLfQdfUVLsMypfHOox6Nqc8um29vd6fqUFi6CYyxt5jR5IOFOQJPzFdDqkniFbhP7Fg0+SAoCxuncNuyemO2MVyfizw9Do/wANHsZJGupLjULaS7nfhp5HuE3sfr+gxV3VNI1nw74a1G50fxBJb21lbTXEFr9kiZVCqWC5K5I7ZzmnZdBGho2u6vN4qudE1q1s45IrNLpZLV2IIZ2XBz/u101cV4b8ParJeQeI7nxBNNPfW0Pnxm2iCsigsFGF4GXPSu1qJWvoNBVe3/5Dk3/Xun/oTVYqtB/yHJv+vdP/AEJqz/5eQ9f0Za+GXp+po0UUV6pzC0lFFABRRRQAUUUUAFFFFAC0lFFABRRRQAtJRRQAUtJRQAUUUUAFFFFABRRQaAAGjIqtfX0OnWb3NySI06kDNcdq3jxWRo9KjIYjHmv2+grlr4qlQXvs6aGFrYh/u0L4416WG6Sxsp2TC5l2H16CuHJJJJOSfWnSSvNK0krF3Y5LHqabXxuKxEsRUc3t2PtMLho4emoLfqwNW9P1W70uXfaSlM9V6g/UVUornjKUHzRdmbyhGa5ZK6OoHjq8EeGtbcv64P8ALNdPomoPqmkRXcqKjsWDBenBxXmFdB4e8TDSYDbXMTSQltylTypr2cDmM1V/fy0PFx2WwdL9xHW53+aM1T07U7XVIDLZyZx95G+8tW6+phKM480XdHy8oSg+WSsxaKKKokKKKSgBaKKKACp7WTDbT36VBQpKkEdRQBpZpajjcOgIp9ABRRRQAVlajZS3er6cQo+zQO00hz1cDag/8eY/gK1arTF/t0G1yq4bcuPvdMUAWaKKKACiiigBaSiigApaSigBaxtZ1uTTLiOOOJXDLuyx962RWNrWiS6ncRyRyqgVduCPevLzX619Wf1T47rt+p04X2XtF7XYXRtak1SaRJIlQIoPymqWparquk6zp8Us9pcx31yYkto4WWRUClt+dxzjAzxjntVzRdFk0yaR5JVcOoHArK0zS9dg8QXeqanZWd3cTO0cMv2xgIIM/Kip5fHYnnk1WW/WXho/Wvj1v9+mwYr2XtH7LYzYfGOp3HhCfUm1DTkvFgRzaRwFZICzgZIZjkDJ7Uah4n1PTl1aODU01A2tpbTxukceVZ5SrLxx0Hf1rYm0rUdX1u1u76wtbSGzilUqs3mmcuANp+UfLxnnviqFnDLc+F49O1bw7fRXMqRi6ktY4FBdWDZGH6ZHpXpb/wBepzFzVdU1i0Gn3NvOi/bbqKJLGaAb8E5fkN2UMfwqTRNXudaj1a1+2mC6tb6WJJBCPkRXIUcjaeB9a1YNEt01Y6nM0s9zt2RGVsiBf7qDoPc9T61laN4bhlhv01zToZd+o3E0ImVX+VnJBHpkYo/r8ge39eYfadU0/wAV6ZYzan9tgu4pnkVoUUrs24IKgd2qHUtb1Zptcn06W3hg0UcxSR7jORGJGyc/KMHAx9eelJ4Y8HadZJdSXujWqTi/nkgZolJCGQlCD2GMVrX/AIX0zUrqWe5jmDTqFnSOd0SYDoHUEBuOOe3FHRB1ZgzeMZ411siQeasUQ062Cgu8rwhgoHVvmP4D2p58XXUzeHvsaNLLdSvDfW21VkjcJkqQfukHmp5NK1GDUPEtzp9uqTyxR/2fI23G8QhePTkY5qppVlNJ4h02a1t9TxErm9fUXLhWKgDZuJw2c8rxj8Ka3/r+vX5C1sv6/ry+Z246UUUUhhRRRQAUUUUABOBmvHPH3ia01fVtGvNOtr6VtPmkEsZtmGVYAZB/CvY6b5af3F/Ks6lNVYOEtnoXCThJSXQ8OutR0W+nWe70q6llUABntGJFWvDviGysPHk+r3dvfrB9iEEeLZjklgT/ACr2fy0/uL+VHlp/cX8q8rCZRRwlRVISk7Kyu7pHVVxk6seVpHI/8LN0P/njqP8A4BtWP4t8caRrfhLUdOt4dQ86eEiPNowyw5A/MV6N5af3F/Kjy0/uL+Vexqcl0ZfhVy/hHSiwYEWkYIYYIIUA1c1G1N1aER481GDxk+oOatAYGBwKKmcFUg4S2YRk4y5kZS6nbhcXDG3kH3kkGCP8aX+1LH/n6j/OtMqp6qD9RSeWn91fyri+rVl9tfd/wTb2lPs/v/4Bly3+nTxNFNNDJG4wytyCKq2qaBZW0lvaR2cMMn3440AVvqO9b3lp/dX8qPLT+6v5U/q9b+Zfc/8AMXPT7P7/APgGJYtommRNHpwtbVGOSsShQT+FLNLo9wlyszW7C6j8qf8A6aLgjB9Rgn862vLT+6v5UeWn91fyo+r1v5l9z/zDnp9n9/8AwDh08JeEo41SOSVEUYVVvZQAPT71dFbXmnWtrFbw3KCOJAi7nJOAMDk9a1vLT+6v5UeWn91fyodCu95r7n/mHPT7P7/+AYeonR9Ws/st/LHLD5iSbd2PmVgynj0IBqS7udLvrGezup43guI2ikTdjcrDBH5Gtjy0/ur+VHlp/dX8qPq9b+Zfc/8AMOen2f3/APAMm2vNNtLWK3gnjWKJAiLu6ADAFSf2pY/8/Uf51peWn91fyo8tP7q/lS+r1v5l9z/zDnp9n9//AADNOqWXa4Vj6LyT+AqTT45HmmvJkMfmAKiN1CjPX3JJq/sUdFH5UtXTw0lNSnK9uyt+rFKpGzUVuJRRRXaYhRS0lABRRRQAUUUUAFFFFABRS0lABRRRQAUUtJQAUySaOEAyuqAkKCxxknoKfXOeNPsa6TbveGIMl7bNGZCMgiZMkfhn8KANptQs1uxatdQi4PSEyDee/TrU0c0cpYROrlDtbac7T6GuW1R4j4m0+7tbpbmZWWEWe0EKrDLSgjkHGOemOO9SeHHtxea9DpMlur/bd6hcEDMceTge+fxoA6imtxUIF13eL/vg/wCNV783a2M5WSMERnGEPp9aipLkg5dhpXdi2k8bsVSRWK9QD0qTNeb291dW1wJopAHB7g8/rWze+M5NPaNJYVdniV8gYHP414OAzyjioy5/daPRqZfVUlGnrc0fGUix+GbgN1Yqo/OvLa2dd8TXeuKkcipFEhyFUHk+9Ynzeo/KvNzGvDEVuaD0R9DluHqYajyzWrY6im5b1H5UoORmvOsencWiiipGJRS06IhJkY9mB/WmtWJuyudP4N068XUPthVo7bYQSeN/sK7WlH3VxwMDijFfd4TDxw9JQTufB4vEvE1XUasFFFFdRyhSUtFABRRRQAUUUUASQy+W3P3T1q6DkZFZ1WbWT+A/hQBZooooAKry/wDH5D9G/pVioJf+PyH6N/SgCeiiigAooooAKKWkoAKKKWgAFFFUb+/a0kVVQNkZ5rOrVjSjzT2KjFydkXqKo2F+13IysgXaM8VnRavqQ8az6Vcx2wtPsf2iEoGMn3ivzHOO3QCinVjUjzR21/AJRcXZm/RXn8PjHU7jwhPqTahpyXiwI5tI4WWSAs4GSGY5Aye1GoeJ9T01dWjg1OO/NraW08bpHHlWeUqy8cdB39a02JPQKK5TVdV1i0Gn3NvOii9uookspoBvwTl+Q3ZQx/CpdF1i51qPVrX7aYLq1vpYkcQj5EVyFHI2ngUf1+X+Yf1+f+R01Fcz9p1TT/FWmWE2p/bYLuKZ5FaFFK7NuCCoHdqg1LW9WabXJ9Olt4YNFHMUke4zkRiRsnPyjBwMfXnpQG51tFcTN4xnjXWzvHmpFENOtwoLvK8IYKB1b5j+A9qefF1zM3h77GjSy3UrxX1ttVZI3CZKkH7pB5p26Cvpc7OigdKKQwooooAKKKKACiiigBGYKpZiAAMkk9K5K6+KPhO0umgfUw7KcFo0LL+dVfixfz2fg5Le3kaM393HbO6nBCnJP/oOKwLXS7O0tUggt41RRjG3rXiZrm8Mu5U48zl8juwuEeIu72semaZqtjrFmLrTLqO5hPG+M5wfQ+hq3XlPgFf7L+Jd/ptn+7tLiy+0NEOgcMoyPTqa9Wr08NXjiaMa0dpK5zVabpzcH0CgkAEk4A6k0VR1cn7CI84EsqRtj0Lc1dWp7Om59iYR5pKIh1ZHJ+zW886/30UBT9MkZo/tN/8Anwuf/Hf8alVQqhVGAOABS1w89Z7z/BG3ufykP9pv/wA+Fz/47/jR/ab/APPhc/8Ajv8AjU1UNT1e20tUWXfLPLkQ28I3SSn2Hp7nAHc0c1b+f8F/kHufy/mWf7Tf/nwuf/Hf8aP7Tf8A58Ln/wAd/wAawNS1rVbO80VRZrvu2l862SQHAC5HzEdu9ZcnijUzA07xyKU1xbXyItrMY/LztzwDyc1X77+f8F/kK8P5fzOz/tN/+fC5/wDHf8aP7Tf/AJ8Ln/x3/GuPbxDdSanaNpWo3V5HJdPHPBLZbI4kCvkGUJgYZQucn8aq6d4i1TVddl0+e/j0+J7mfbNC6Sj5FixEjsuP42bpnjjoaP338/4L/ILw/l/M7r+03/58Ln/x3/Gj+03/AOfC5/8AHf8AGuTt9dv5I7CM3QnA1trI3KqALmMRyHPHHUAHHdTW7JrYstRNtqsBtYpH2291nMcmegJ/hb2PB7HtSvW/n/Bf5BeH8v5l/wDtN/8Anwuf/Hf8aP7Tf/nwuf8Ax3/GpqKXNW/n/Bf5D9z+X8yD+1SvL2Vyq9zhTj8jVyCeO5hEsLBkPQ1FVazHlatcxpwjxpKR/tEkH+Qq6dapGcYyd09BOMZRbStY0u1FJRXoHOLSUUUALQKSigBaBSUUAKKDSUUALQaSigBaKSigBaKSigBaY0Sv94KfqM06igBnlKDnaufXFKkSISVVQT1wOtOooAWo5cbTu6Y5p9I1J7Aee6kYPt8ptf8AV54xWV4kU/aLM4ODbKOnua9Cc6Kb2SJ3tPtEa+ZJGWXcq+pHYe9UtcGmyw28uYJJwhe2Un76jGSB3xkfnXxyyWph/a1pSWutkvme7QzCKnBW20POJbK6ghWWa3kjjbozLgGq9dVFfrdIzOVuoZeGXOVPPajVdLstMtVV7CZo3+Z5Q/MZPQA9683D8mJUnT05d7/11PaWMcWozWr7GBaabc38btbqCF4+Y4yfQepq41vpll5dvdrNJOAPMaNwFBPbkdq5rW/GAtmvbLTIWWPT4RKgBy5dgcs3I6YHSsbU/FE9vfiKJoySq7dwzvf+IE7hjH416MaFo2grvu9iJVXJ3nKyXRb/ADOy1K2Wz1KaBM7UbC59O1Va2r+yOpMb21nt5U8hW+WTJcKoy2Kxa5MRSdOb00OvD1VUprXXqFKil3VUGWY4A9TSVJbTta3UU6AFo3DAHocVjG3Mrm0r8rsetDhVz2Apa4iDxvdG9VrmJBb9GRF5+ua6qw1ay1Jc2k6s3/PNuG/Kvt8PjaFb3YP7z4fEYKvQ1nEu0tJ0ortOIKoa1efYdLluBd21mVx+9uVLIPbAIJPtmr9U9Vs2v9JuraJxHLJEyxSEZ8tyCA34HmgDk7fxjqqCBb2wTzZrGe6HHlKCr4Tdub5Qwx64JqvpHjy7vb61t2ks5xPdJEXjG1gCjsw27ieCoG7oc1qeHvDV1Y6lJcXkFvDE1ssJijlaXzWByXO4DH0qnpvgy5t7zTZby108vYymTz4GZGYndk7enO48UAdx2FKKri2Adm82U7jnG84H0pfs4/56S/8Afw0ATGlBwcjrUH2cf89Jf+/ho+zj/npL/wB/DQBqQyiReevepTWOsO1siWUf9tDV8WwI/wBbN/38NAFmq8v/AB+Q/Rv6UfZR/wA9Zv8Av4aFtlWVX3SMV6bnJoAsUlFFAC9qKSigBaKSigBaKSigBaoX9g93KrK4XAxzV8UVlVpRqx5Z7FRk4u6KNhYPaSMzOG3DHFY72Wt/8JwdUSxtGtBafZgTeEOfnLBseX79M101FOlTjSiox2X6hKTne5y02lajq+t2t3fafa2cNnFKpVZfNM5cAbT8o+XjPPfFULOGS58Lx6dq3h2+huZkjF1JbRwKC6sGyMP0yPSu4orQkzINEt01Y6nM8txcbdkXmtkQL3VR0Ge56msvRvDkMsN+uuadDLv1G4mhEyq3ys5II+orp6KAOR8MeDtPs0upL3RrVJxfzyQO0akiMyEoQewxita/8L6ZqV1LPcJMGnAWdI53RJgOgdQQG4457cVsUUB1ZycmlajBqHiW50+3VJ5Yo/7PdsY3iELx6cjHNVNKs5pPEOmzWtvqWIldr19RcuFYqANu4n5s55XjH4V29FC3DoFFFFABRRRQAUUUUAFFFVtR1G10nTpr6/lENvCu53PagDj/AIt2ctz4OjuYUZxY3cdy6qMkqMg/+hViWuoWt3arcW86PGRndu6fX0qLV/GWteMEktNBiOm6U4KvdSj95KvsOw/zmsmLwLpEcaKRMxUckyEbvrivh+IauBrTjGVR80b7K/36o9zL4VoRbUdGbXgQ/wBo/E6+v7P97a29j9neUdN5ZTgevQ16rXi+mDWvA8zy+HWF7YO26Wxm6+5U+v8AnmvSvC3i/T/Fdo72m+G4h4ntpRh4z/Ue9fRZVXw1TDRhh5XUVbz+aPPxdOoqjlNbm9VDV/8Aj3h/6+I//Qqv1m6tKh8iAMDKZ0bYOTgHk12YtpUJXMKKftEWaKKK5ygrO1TRoNTaKbe9vdwZ8i6iOHjz1HuDgZB4OKvyOI4mcgkKpJCjJOPSsTw74oi8Q3WowR2V1ZvYSrG63KhWO5QwOATjgimr7gQeI7WVvCyyajZ2+p3VvLGxxblxtMiq7BOTnYWPFQ+ZoN2bG0so7m1EN2lwiR6fNGrOARySmAOetXvFXiqy8JaYl3fK8rSSLHHDH95yTyfoByT/AI0ninxXY+FNKjvL5XkaWRY4oY/vOSeT9AOSf8aauIbb+GZ7OOWCz1u8itpJZJPI8qFgN7FmGShOMse9YrWTafpOtf8AEtE8Y1iJhG9vv3REwrI6qBzhd549Kt3vjiWF9SlsdFnvbHSnZLy5WZVKlVDPtU/e2g+oqg/xKuG0fUNZs9Ae40uxmkja4+1BSwRtpYLtppSFoa73WjalPplvbfa4Psl0JokFhMik7WXBJTAHzmr1xoj6nfmXV5/OtY3DQWajEYI5DP8A3jnnngccZ5ptp4gW48WX2jOix/ZreCZXL8v5m/jHts/Wudl+IuoRaDJrn/CN79NSfyTIt6DIf33lZCbP73bNFn0Gd5RXOeHvFn9va9qenjT7izWxigkBuV2u3mb+2TwNn610dQ1YYVWg/wCQ5N/17p/6E1War2//ACHJv+vdP/QmqP8Al5D1/RlL4Zen6mhRRRXqnMFFLSUAFFFFABRRRQAUUUUAFFLSEcUAFFYGt6rd2GtafBEyJbTtiaSSFiqDOPvA4BJIAzSWHiB7zxPc2Uca/YoodyzE8u4ba2PYdPqDQB0FFIGBHUUbh6igBaKTcPUUbh6igBaKTcPUUbh6igBagu2mSBjbKrSAcBulTbh6ijcPUVMlzRavYa0Z47Lpl/f6xqVlJeQWxtnmuwHUJK7zRuAm4nlfmP5CkvvCUf2ywa9v7eG8t9PDwJIgbeuFV8vnqCBtx0x3zXqk+j6feSCW9tbeeXG3e8YJxUTaXZ3rSR3lpbTxwNshDRA7F2jiuOODtH3pOUrWu/8AI2dZt6KyPNPCNu1ho1pHc/P5bsW+YNuG8nqK1rs26zahBf3jZeRHOF5K9cDsDkitbV9N03w7Ek6jbHu/d2ygAE9fyrh7m4e6upJ5PvSMWNfHU6dXBTqKqleTufS4WCxKUo6JdfMq6v4JUpNb6fbO4lD+RcFSS28dSe/WsPXvh/qk+pEyrEBKAkWIwWztAOGwa6w6pe+SkQuZFjRNgUNgYot9Tu7WJ44ZiFfr7e49K6o4mEXdX/yOqWHqyjZ2LDj+x9Bh01ZF+04xcbOeMcKTWZSSSBVaSVwoHLMxwB+NV49Qs5ceVdwvlto2yA84zjj25rlqSlVd0tEdVKEaUbN6ssUtV1vrR/8AV3MT/MF+VweT0FWKxaa3Nrp7BU9hBLc38MEDbJJHADZxioKnshIdQtxASJDIoUjsc1VO3Ork1L8jt2PULGCS1s1hmuHuGXrI/WrFHfB5Pc0V+gxioxSR+eyblJthSUtFUSFFFFABRRRQAlLRRQAlWraRmyp5AHFVqcjmNww/GgDQFFIrBlBHeloAKKKQ8UALRUS3MTyvEkiNIgBdAwyuemRUb6haRxtI9zCqK21mMgAB9CfWgCzRTUkWRQyEMpGQQcg06gAoopaAEqGeZo2G3HIqao5IPNIJOMUAMgmaRyG9Kw473U4vH09hNeCWzex+0RQiJR5Z3levU9PWt+KDy2yDnisKTRdYPi46wl9YiL7P9nWE2zltu4sCW39efSl1+/8AJh0fy/NHMjXNUXwpL9u1TUI9aa2DtbSWyx4+dd5QqozgE9zimX+r3Vtb6ydI1K9uYIrS1eOSWR+JWmIYBiO4xnHrXVHRdTvtXtr3Vbm1Is0kECW0TLuZwBuYlj2HSqVrZa43h6DStZ0q2vdqIs8xvyDKVIO7/V56jPWnv/XqAzV5tRin0v7PqFxDqF7cR5slkDosYIMh6ZxtGM+pFM0rXIbiHV7XWtRlswmozpFcSXAiBVXOFRs54HaunttJs7W+mvY4R9qmADysSzYHRQT0HsKpaDo8mnx363qxP599NcJj5sK7kjqOuKP6/IHt/Xmc/FrGn23i/SoNO8S/a7eZJhOsl8JV3DYFHXg5J4+tJrl7qRk8VT2+qXFuNLhV7eOPbtB8kPzkc8mug8O6I2l294tzHCWmvp50KDOFeQsvbrg1JfeF9G1O6lnvrCKaSYBZCw4cAYGR344o6IOrONv9Yvj4qMFxqN1BYefAJmiZhsU25bqOmWxT7TWNUWfTFtZ7i7jk1i4hQTyMnmwhGK5JGSBjI47Vs3/h7VBca5PpckUUl2bY237wrgR7dwJxxkAjv1rQjhv72/sn1HRreMWrl0lS8LFGKkZ27Bng+tNbi6f12NwHgZ60tJS0hhRRRQAUUUUAFcB8YJG/4RiwtusdzqMUcg9VwT/Su/rzb4u3a283hxZpAsH2xpJM9PlAwf1NY1240pNb2ZpSV5pMjRVRAqAKoGAB2p1ZP/CT6N/0EIvzo/4SfRv+ghF+dfj31TE/8+5fcz672tP+ZGtVPwwy2fxcUbtgvNPYBR/EwIPP4A1V/wCEn0b/AKCEX51s+Etb0C58SR4nt5LxkKQtjLA+gP0zXt5FCvQx0XKDSd1szixrhOg0mjvdRS7e1xYuFfPPqR3APY1TsDbKWRFaOfrIsv3z757/AFrXqlqdqZ7cSQoDPEwdD3ODyPxHFfe4mg+b20dWun+XZ/meFTmrcj/r1JKKrw31vMmRKqt/ErnBU+hFSfaYP+e0f/fQrBVINXTG4yWjRITgZPArkPDMsY8deMSXXH2m3Oc/9OyV087WlzbyQTvFJFKhR0ZhhlIwRXOTeCPC62tyNO0/T7W5mheJZ1QZXcpUngg9CatTh1YuWXY4vxtPqsthr+oaroN4QSlvZTK8RiigEqHd9/OXIyeP7o7VH4tudYuNG1rU9Z8N38TMsUFmxeIpbwiRD2fO5yOePQdq9NttL0uLw5baLO0VzaQQxw7ZmDbwgGCfyBq1eRafqFm9reiCeB8bo3IKnByOPqBV+2ghckjykaRc+KtI8Ua0mpS6LEZpFm0wMBGxhGG8/nqwXnaRxjrVjQtGm8Q6LdaLb6ubG21SBdTns1tFzEtwxbaHJ7EEdOmK7m+8KeGNSvmvL2xtJZnIMhJGJMdNw6H8abqHhjTL7Vf7Ri1C5sbgwLAzWd2YgyKSVBAPbcaPbQ7hyS7FS08PpL8TNS1O9sFkSOztVtLiRAdrjzN209jytczrvhue2WPwjZ399qMF2z3v9noIYlSMTByXlKFsbyAMc13ek6XBpU0kg1e8u96423d4ZQvuATxUWs6JZ6tfW9/DqUun39ujRpc20ihijEEqQcgjIB5FJVY33DlfY5/wDp8Fn4q8QlLRtOn8u2jlsmfzNmA5Dh/4g270GMGu+rE0PR7LRHup/tz3l5eMGuLq4kBeTaMKOMAADoAK1/tMH/PaP/voVMqkW9xqL7ElV7f/AJDk3/Xun/oTU5ru3RSzTxgDvvFN08Ge6nvCpWN1WOPIwSoyc/maiLUqsFHXW/4MqzUJNmj2oopK9Y5RaSlpKAFoFFFABQKKKAAUGkpaACkPSlooAydT0mbUz5Ut0y2pZWeEKPm2kHr1xkCq9v4S0211P7XBAEHkGDyh93G7OcVvUlAEIs7cD/Up/wB80v2S3/54p+VTUlAEX2S3/wCeKflR9kt/+eKflUtLQBD9kt/+eKflR9kt/wDnin5VLRQBF9kt/wDnin5UfZLf/nin5VLS0AVzaW4H+pT/AL5rm9Y8WRaHqL2cFmHwAzENgZNdWRkVlXugWN1bXKtCDJPklzyc9ua5sRGtKH7l2Z0Yd0lO9ZXRyV14xsNUg8rU9MLr2IfkfSuXuxa+dmyaTyz/AAyDkfjViHRb65vprW3gZpIc7geMVVuLWe0lMdzE0TjswxXx+Iq16qvVXzsfY4elQpS5aT+VyIUUCiuI7iO4V2t3WEqHIwC4yPxFcXZ24bWrC62hg9xKxfC7pXCMpGMfKMds9q7g1XjsLWK4aeOCNZW6uF5NdNGt7OLXcwq0vaNPsc/c2UkTR3k8awSXF5CBEnRFBOAffkmumJOeBmhkV8bgDg5GR0NOqJ1ee1yo0+W9hvPp+tbPha3efxBCQm4RZkIz6D/Gsiuz8D2Dxwz30i4EmI4z6+prpwNP2uIireZy4+p7LDyd99PvOm8yb/niP++6PMm/54j/AL7/APrVKKjmuo4JhGylum456V9nUqQpq8mfExi5aITzJv8AniP++/8A61HmTf8APEf99/8A1qnKndgc+lMMkSyCNpVD+lOU4xV2xJN7DPMm/wCeA/77/wDrUeZN/wA8B/33/wDWqfYwGccCm5qxEPmTf88B/wB9/wD1qPMm/wCeA/77/wDrVNRQBD5k3/PEf99//WpfMm/54D/vv/61S0tAEHmTf88R/wB9/wD1qPMm/wCeA/77/wDrVNRQAtvczqdphHPT5/8A61WvNn/54f8Aj/8A9aqmcVehk8yMZ6jrQAzzLj/nh/4//wDWpBLP5iK1vhDnc2/OKs0h5oA4rU5FTWvES6U6JeyaYgUxj5i4MmfqQCP0qvFqqaZpz2iLBdoGjjtZniCIJGUltx6fKBk9+cda7ryY9xbaNx745pPIj242KRnOMUAUtCtorPRbS2gm89I4wokB+971pVENsY4woH6U5XDDKkEHuKV1ewD6KSlpgFApKWgAFBpKKAFoNFJQApooooAKKKSgBaKSloAKKKKACiiigAooooAKrXmm2Wobft1rDcbPu+agbH51ZooAzP8AhGtE/wCgVZ/9+Vo/4RrRP+gVZ/8Afla06KAMz/hGtE/6BVn/AN+Vp8Gg6TbTrNb6dbRSIcq6RAEfjWhRQAUUUUAQy2lvM26WFHPqVpn9nWf/AD7Rf98irNFZulTbu4r7ilOS2ZW/s6z/AOfaL/vkUf2dZ/8APtF/3yKs0UvY0v5V9w/aT7lb+zrP/n2i/wC+RR/Z1n/z7Rf98irNFHsaX8q+4PaT7lb+zrP/AJ9ov++RR/Z1n/z7Rf8AfIqzRR7Gl/KvuD2k+5W/s6z/AOfaL/vkUf2dZ/8APtF/3yKs0Uexpfyr7g9pPuVv7Os/+faL/vkUf2dZ/wDPtF/3yKs0Uexpfyr7g9pPuV1sLRGDLbxgjoQoqx0HFFB6VcYRj8KsS5N7sSiiiqEFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUlAC5ozTSfSsu7vLq2uCONh+6cVjWrRox5pbFxg5uyNMyKCFJAJ6DNOJyK5qa6kmm8wnBHTHatmxvluIwGOJB1HrXNRxsKs3HbsaVKLgrj5RFaxzXG1VO3c7Y64FeRahqFzqNw0t1K0hzxnsK9ZW9t7pngcdflww4auF8V+GG09pL+2KmB3/ANWq42Zrz8zhKtSUqT91bnq5TUhTquNTd7HLUUU+GCW5lEcCNI7dFUZJr5lJt2R9U9FdjKKkmt5bd9k8bxt6OuKioaa3BNS1QtGa3tL8NTzvb3N35a2r4cgt8xXqOPeuiWw09ZFcWMIZDlSox+frXXHCtq83Y8+rmFOEuWKuY2iaHaXOlCbUI2zJJmMqcHaK6e3u/s6+WsY8kDCoONoFV2Yt1+gGOlJXZCo6VvZ6WPFqylXbdT7uxebUf+ecQH+8c1UmlaeTe+M9OBTKWqqYirVVpsxjTjHZDxPLs2+Y2PTNRnk8miisXJvcuyWxpx3VvFbqodie+ByanUh1DR/MrDIrFqRJ5I12o7AegNenRzCUNJLQ5Z4dPbc1qKqxXsjqd8asB1bO2pRdW5YIZUWQjOwt/WvUhjKM0ne3qcsqUkSUuao32qxWE8cToXLDLEH7op11qttbW4kV1lLD5UU9fr6VnLMcLFzTmvd3GqFV2ajuW6WsqDX7SRf3yPER6fMDUkGuWlxOYyGi/usx4P19Kyhm+BnZRqLUt4Sur3jsaNWbZgF2nhieBmsjU9QGnwjYA8r/AHfQe9ZGlam0WrCe8kLBxtJPaufFZ1h8PiI4fdt6vorl0sHUqU3U6fmdvRTVYOoKnIPIpa9zc4xaQ0tJQBy+taXqV1qQMUrNbyEcBuE/CuitYRb26RJ91AAKlxSgVx0MHToVZ1Ytty79PQ2qVpTgoPoLRRRXYYhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABS0lLQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFJS9qKACkpaSgBaKKBQAUUUCgBKWgUGgAoooNABSUtFABSUtFACUtQ3VzFZ2slxcNtijUszHsBUemahDqul21/bBhDcxrIm8YOCMjIoAs1XvJzBbM643DoDVmsPU0ndzK64jXgc1y4qo6dNuK1NKcVKVmXYLz7S+Y/lRBlyfWodTu4xGYQNzMAc+lZ0NzJBA6KOJO9RM5fG7nAxXk1Mc5UuXq9zsjQSnfoNpQSCCDg+1FFeYdVhVZlcOD8wOc1tLc293Z7bhozuXDq1YlArpoYmVG/VMyqUlOxRPh/SI5XxC8oLZG5yAPyqe2srKxkMlnarFIRjfuJI+masUlY89neKS9EbuU5Kzk38xk8cV5EYryNZUPr1H0NZ9v4XsrVmmnP2pHOI1PG0e/vWmalhdSpikOFY5B9DVQak7T1fQn2k4Rag7Ii4EaRooRI12qo7CinSRtE+1ue4I70wnC5qJc3N725KtbQWg15bca/qQ8Zgi/ukOdq6btG5k3Z3Zx19sdO9HiPxRqtrrN9a2t/NAtxxboxVTGQgB6qT1yeo61usNJtK5PtEj1KiqVpeBrG2kkEmZEBOVOc471Y+0R/7X/fBrn5WXclopivuztV+OvyGh57eBA920kYZto+Q/nT5Gld6IXMr2H0u1RGZZnEcY43H+lRNKikfeKn7rBTgiq+qyRvYQsWZXUkKpU/MPWs6zlSpSqWvZDjaUlG+5FqGoLOggtwViBySerGs/Jz3pnmL/tf98ml81f9r/vk18bXq1q8+eZ68IQguVDiSxyxJ+tFM81f9r/vk0vmL/tf98msHGb1aNLodQab5q/7X/fJo81f9r/vk0uSXYLoeWLY3EnHTJqSBQ9xGp6M4H61B5q/7X/fJp0c6xyo/wA3ysD901pBS505dyZW5WkekRgKgA7CpKpw30UkKON5DAEfIf8ACpftkfo//fs/4V+0ws4po+Oe5PR+NZ15dwyWNy73ElpHEPnmxt2gcnGa58T6mNBsbprueEtqChQ4G6SFpMKH4/ukVQjsqSlHSigBKWiigBKWigUAJRSig0AFJS0GgAooNFABSUtFACUtFFABRRRQAUUUUAFFFFABWD4l8Yad4WktE1FLh3u93lLBHvJ24z/MVvV5p8T/APka/Cn+/P8AySsMRVdGjKouib+40pRU5qL6mp/wtXRf+fLVP/AQ/wCNH/C1dF/58tU/8BD/AI1y/wDajf8ACT/2X5S7Ps/nb++c4xRc6o0HiKz00RKVuI2cv3GK+R/1ixLaXsltzb9Nz1/7PpfzdbHa6R8RND1fU47BDcWtxL/q1uYim8+g966qvINX0u61G40z+zYd9zDfRyAggFVGSTmvXx0Ga+gyvMPr+H9q1Z3tY87FYf2E+VMKZNMkELyynaiDJNPqhq/NrEp6NPGCPUbhXoVpunSlNdEYU4qU0mMF3qE3zxQwxIeglyW/HGMUvm6n/wBOv/fLf41Yorz7Te82b8y6JFfzdT/6df8Avlv8aPN1P/p1/wC+W/xrI03VrvX7wy2RW10+3lKOHAM0rDjBX+Ad+eTx0rk7i91YWl/YJPJvbWHKQ/Ypi5j+17hiQfKBt5+lUoT/AJ394uddkehedqR6G14/2W/xoM2pjqbUf8Bb/GuIuLuaHXNehmtBd21xq0EKIJjGwkNpCRz2XANVdXt9S0nQbmO4aW1judSieK3TM4Ee+EYL4+XkMfxp+zl/O/vFzrsj0LzdT/6df++W/wAaTztTPQ2v/fLf41yd1ZWt/q8MugQTA2srT3N3CPvEK2I03cMSxB9OKr6GltNZa5d+IZrpFttRcs87CNxmKLgiM4z0wB/M0uSX87+8fOuyO087Uh3tf++W/wAaXzdT/wCnX/vlv8a4YSXukWnhiW5hnlkNxczNCz5dEZHcKSepVSB+FaNxN4KluDJdQxGadycvbSbnbqf4eT1NHJL+d/eHOuyOo83U/wDp1/75b/GjzdT/AOnX/vlv8a5nwrLd3XgXT00y48m8t41MiXELYPGfLbIB79R0Ira0LWl1q1lcQmKSCUwygMGQsAM7WHDDnr+FJwmvtP7w5l2Rb87U1522rewDDP45q1aXa3cbHaUkQ7XjPVTSVWtuNauAOA0CE+5ywp05zhUiuZtPTX0BpSi9NjSpKKK9M5haSiigBaKSigBaKSigApaSigBaKSigBaSiigBaSiigDN1zTJtVtoYobjyRHKJGUruWTHQEemcH8Kq+HtIu7HQNPtr+dvPt4BGwiYhf/r9OtblFAEH2b/prL/32arTaZ58g3zy+X3Teea0KKmUVJWY07HP6rY+U0ZR32YwBuPFZ3kf7cn/fZrpb9EMW+Rd4Q52+tQz6dFLBujUJJjPFeNicJOdRygzsp1kopSMHyf8Abk/77NHkf7cn/fRqXGDikryOZnZZEfkf7cn/AH0aPI/25P8Avo0/eu4JuG7GduecUEgDJOBjqaOZhoM8j/bk/wC+jTorXzZlTzHGTjJenDmnKMyKCccjn0qov3lcT2JrvTBFAzwSyFk52s3WljiWCLfcD5iPlQjr71kah420rbFEfOzcTPCNuAwMcyxE9em49fSqWp+PdOHiZNHNvc+aPlEgMezrj727r3x19q9nE4WEI+0gtexxU6kpPlbOgll83aAu0KMAVHSClrxJScndnalZaHOyeEYH1Frg3Moie5F00W1fvg5+9jcBntmtLUdFsNTtp4rq3RvPjZGfaNwBGOD61fpyxs5+QFiPSq55yaCySI7eDy4Y4YgSEAUVaS1YTLu+ZM8svIp6ovlSxRylSv8ArGXgocfyrmUuZoW/dSsuDkYNcuKxUMG4ucb38+xVKnKtdRdrFnUby7e4eGdyAjfdHSq0lxNMiJLIzqn3QTnFMkdpZGeQ5Zjkk02vka2IqVJylzOz79j1oUoxilYswahc28eyKQhewPOKjnuJbqTfOxcgYHtUdFRKvVlDkcnbsUqcFLmS1Cj8KSisTQWj8KSloAKPwpKKAF/CikpaBHZeG7gz6UFY5MZ2/hWz2rD8M2stvYMZV2+Y24A+lbdfr+VObwVP2m9j5LE8vtpcu1yrqWmwapZta3asYmZWO1ipyCCOR7iqtx4ftrq1jt5pbpkicSKftD5yDkc57Vq0V6Rzgo2oBycDHNFFFABS0lFABS0lFABRRRQAtJRRQAtFJRQAtJRRQAUtJRQAtFJS0AFFFFABRRRQAV5p8T/+Rr8Kf78/8kr0uvNPif8A8jX4U/35/wCSVx47/dKv+F/kb4f+NH1MK/0vS9Q19POmkW+WIMFRyPkB/wAasan4ftNVuIprkyrJEpVTG5XiqOo6fqq+KE1LTEgdPIELeaxHGcmuiHTmvyyrWqUo0pU6l9O+3deR9PGMZOSlHr95m6Jc+JNUuHvPC1lbSWllIYxJcuQZmA5xivQvCfiRfEuktO0BtrqCQwXMBOfLkXqK4L4feM9F8M6Hd6VrV0Le4trmR+AWDgnPBFb3wszc6dq+qLgQ6jqMs8S5BIXpz6Hiv1DAYWjhqKjRWj19fM+axFSdSTc+h3dUNX/494f+viP/ANCq+SAMk4A71lXlyt80cNoDIElV3kH3QAc9e5rTGTiqTi3q9iKKbmn2LlFFFYDKEuj2r6nHqEYaG5XhniO3zF/usP4h9aoTrrVp4iurm0to7yzngiVEe58vy3UvuONp6grz7VtXFxFa20lxcyLFDEpd3Y4CqBkk155/wsTU5bHWbhbOG1WC8tYLVpwcRxTBSJZefRgccYziqSbEdZBpMWpRXf8AbGkQwm4nWd1E3mb3VAobOBggKB+FV9W8KWjaPNDpFpDFcyPF854yqyo5GfotZlpreuf21faI2p6dczR2S3aXwh+SD58FZFDY5UEjkVlJ4s8TT6ULuG80+S3utTgsbO5jtyN4ZiGcqWPGenrjPenZiOsZNasNcvJbS0iu7OdI/KRrny/KIBDcbT161LY6TFczNd6jpUVvceeZgom80Fiqjf0HOFArm7zXfEGkzeI7K9voLiWx0Nr+3mjgCbXxIBkc55QGtjTtV1DxH4as7vQL6CKfaouHu7R8M20E4GV7nr0oaYzR1PTprvWtHuotvlWcsjS5POGjKjH4mm6rpct5rGjXMKp5dncPJLng4MTqMevLCuRuvF2t2vhzWZ7q5tRLpurwWYuoYtqMheISZDE4xvYZ9q6m8i1PVDFdaBrsENo0f8MKTBzk5Ib9PwpWaA0NQ02HUrYW87OsJYM6Rtt8wf3Tjt6jvViCCK2hWG3jWONBhVUYAFcxol9rMXja80XVb6O+iisYrlZFgEZVmd1xx7LXV0npoAVXt/8AkOTf9e6f+hNViq9v/wAhyb/r3T/0Jqz/AOXkPX9GWvhl6fqaFFL2or1TmEopaSgAopaBQAlFLQKAEopRUU91BbKDczRxAnALsFz+dAEtJURu7cTLCZoxKwyqbhuI+lTZyKAEopaKACkpaKAEopaKAEopaKAKt+wWzlLelJYzefaKx69DU8sSzIUcZU1Hb2y2ylY84Jzg9qwcZ+15ltYu65LdTAuV2XcgHQMahbO045NWLyN0u3MikZJIPrUBr5eqrVH6nqw1ijyO7M58aNctA/2sT+WuJD9nBLD92Tt+8fr1/KtbVLzUbK08Q2EK+dCfNkmcREmHzE3Y3bh0DDnFdbc+H9PE8l75DM4k88xeYRG0g5DFemc96feeG9K1GaWe7so5JJ1AlJ/j4xz68cV0+2g7NrQz5GWtIiMOj2sZleUiJfnc5J4qxPG8sDpDL5MjKQsmM7SRwadGixxqiDaqjAA7CnGuNvW5r0scbrHgCbTrqwGjCa6hWQPKXYEqxliZmz152s2PXNUp/Dd7J4kYbb4utz5SQkuVMO7Jl8zOM98fhivUrSZZYFUth14O49adNOlujYdTJjAA55r6f2tKVG7eljy+SSnZGQi7FC5JwMZJ5p1WkeCaYbozvfjrwDUEsLwkeYuM9K+clTsuZao9KMujGU9H2BlYZVxg84plIazTaZTSa1KV9A1p+/tXfy5QVfJyQfQ1mVr6xOyRQW6nCsm5lHc5rJr5PM1CGIcIbLp0v1senhbundhRRRXmHUFFLRQAlFBozQIKKCRRmgAooooGFT2ARtRgEn3TIM/nUFAO05HUVpSkoVIyfRkTV4tHpCgAcDin1XtXLWsTN1Kgn8qsCv2qElKKaPjWrOwUUvairEJRS0UAJS0UUAJRRUUt1b2+0TzRxFjhd7AZPtmgCWiohdW5nMImjMoGTHuG7H0qXOaAFpKWg0AJRSmigApKWigBKKWigBKWiigAooooAKKKKACvNPif/wAjX4U/35/5JXpZOAa8j8a3ur6/qOl3Vl4a1BZdOlc4cDa6tgH+Vc2LhKph5wju01+BtRajUUn0ZlX+uDTfGaQ3lw0dq1twuMguW4rRa0uPEniyDw/BeNZwfZzcXEkY+ZlyBtHp1qN5tTlYPL4UvWYdygNP0K51XTPGM+tT+GdRcNai3jVQMjnJJ/IV8fluWT+s05V6NlFa3tZvo7Hr4jER9nJQne7PRtJ8IaJo1iLa0sIivVnkXczn1JPWuV8UeHP+EOtbjxL4VuGsTAQ9xZg5hmXIBG3sea0/+E6v/wDoU9U/IVl+JPEmoa74av8ATB4V1NGuYSisQMBuoP5gV9vpY8VXudrJEut6PBIkhQSosqlehyM4I7io4p2tXS3u4hESdqOg+Rv8D7UvhlJY/CumR3ETRSx2saOjDlSFAIq5fW32u0aIHa+QyN6MDkGuSvh7t1afxW+/yNIVPsS2/ISiqYvniG26tpkkHXYhZT9CKX+0ov8AnlP/AN+Wrk9tT6uxp7OfYyfGGjalrun29rps9vHGJ1kuI7hGZZ0XkIcEHGcE+uK841aDxANY1+PVkhGl3F/Zx6o1lDICYPJXJTknbwA2MnrXrv8AaUX/ADyn/wC/LUhv4DnMMxz1/cnn9KuOJprqheyn2PN7Cbw3pHiyQaEGPh65sWTUwsLvB5mQI+x5K7ge2OtRX9hoeoaxqF1odnJ/Y1jpbXNwbbfBG9zG2+LaRj5gA2SOxFelreWqqVW2lCnqBAcH9KPtlt5TR/ZpdjAgr5BwR+VP6zT7i9lPscbbaZ4fuvCF/LYXsEOoaxpRgZ7q+8wrvjOFJYk4Bc10Mljqv/CN2Fjot3bwN5SRy3eN+1QoG5B0JPYnimf2L4b/AOgHH/4Cf/WrUivLaCFIobeZI41CqqwsAoHQVLxFL+Yfsp9jkfGOi2mi/DePTrVC0S31rvLnLSM1yhZmPckkk1P4j8Mrp3hnV7/T9W1a1eC1nuIo4b51jjYKzABQcAZ7V0txcWl3F5Vzayyx7g214CRkHIPTsRmnS3ltPC8M1vM8cilXRoSQwIwQeKPrNP8AmD2U+xzfhjwlbrHZa62papJfXFvC87SXrsJcDIDDPIG48e9dlVKO+t4o1jigmREACqIWwB6U7+0ov+eU/wD35ak8RTf2g9lPsW6r2/8AyHJv+vdP/Qmpn9ooeEguGbsPKIz+dTWFvKry3NyAssuAEBzsUdB+pp02qlWPJrZ3f3MGnCL5i9RSUV6xyi0lFFAC0UlFAC0UlFABXO+LryGK3htCkX2m83RxSTAbIhj5nJPoDwO5xXRUySNJMb0VsdMjNAHBC3SC4e2j3SXg1C1NtIQSzQhUBIPpgPn8a7aOa6K/PbKvJ/5af/WqwI0DA7VyBgHHSnUAQedcf8+6/wDfz/61HnXH/Puv/fz/AOtU9FAEHnXH/Puv/fz/AOtR51x/z7r/AN/P/rVPRQBB51x/z7r/AN/P/rUedcf8+6/9/P8A61T0UAQedcf8+6/9/P8A61HnXH/Puv8A38/+tU9FAEHnXH/Puv8A38/+tQZZ/wDn3X/v5/8AWqeg0Ac/q0t35oDxLsPIAf8A+tWfvl/55D/vv/61ausNm4Rc9FrPr5fFv9/I9Siv3aK8hmkjZBEBuBGd/wD9apxwB9KWkrlbNrC0GikqQHx/KxfqEUufwGa8+vfGOsW2naNeTNBI+qFXKxRfcDPGOcn/AGzz9K9Dt3VJv3n3WBVvoa5jVvh/prmK4t0jsVt2BzFHnf8AOjf+yfrXdQjGUEn3OebakYd54x1SC/1cWrC4gtoHe2dYNqyMqsW+YnB2YGfWu/W7a6toXPQoCB+FY7aFqf2jURFLZNY3alfKu4mOzKBWAww68n8a2VtTBbJgoyKAuU6CiqmqXuLfcItc2ogowS2B1pBT4n8uUMRke1ca3N3ojP1sD7XGeN/lgMAelZua3GsLSSNowW8xjkTOeQfT6VSn0hobdpVnR9n3x0/KvncfgcROrKslo9dOh3UK9OMVBsoUUlFeEd4tFFFAFPViRpkuIpZTj7sT7D/312FcVpN1fJqQVrye4hWzuH4cnBZlI27gNwGSM5PTtXfyIssbRuMqwKkeoNZlhoMFjcRyiaWbyYjDCshyI0OMgfkOvpXpYXE06VKUZLU56lOUpJo47w7r2oXevWdrPeTzLBlJEJT5yV68HJ9e9ehszADYu7J55xVaTTbWS5gn8pVkt2LIVGOSMVcqMXiKdacZwjbTYdKnKCabI9z/ANwf99Uu5/7g/wC+qdS1x8y7fma28yPc/wDcH/fVWbC1nvrtIo4sjOWO7oKhrqPCcKi3ml/iLYr08qw0cZi40pLTd/I5sVVdGk5I2o2njjVRbrhRgfvP/rU/zZ/+fdf+/n/1qnor9aSSVkfKEHnXH/Puv/fz/wCtR51x/wA+6/8Afz/61T0UwIPOuP8An3X/AL+f/Wo864/591/7+f8A1qnooAg864/591/7+f8A1qPOuP8An3X/AL+f/WqeigCAzXGDi3X/AL+f/Wrm/E91HNDb6bcJFBc3qMrSyEYgj43EH+9zge/PaurpjxJJguisfcZoA4oafPb+JLO0tQk4F213Jc871jKt8rHp1IA9hXbigKAcgDP0paAFpKKKAFopKKAFpKKKAClpKKAFopKWgAooooAKKKKACiiigAoqtqOoW2lafNe38oit4V3O57CvPW+Kmo3D+bpfhmaa0P3JJJApYeuKyqVqdJXqSS9XYuNOU/hVz0uiuf8ACnjGx8V2832ZJLe5tyFnt5R8yH+oroK0TTV0S007MKKKbJIkUbSSMFRRkk9hQ2krsW46is4ahdTfNbWiiPsZXwT+GKPtWo/8+0H/AH8P+Fcv1yl0v9zNvYy8vvRo0VnfatR/59oP+/h/wo+1aj/z7Qf9/D/hR9bp9n9zD2MvL70aNFZ32rUf+faD/v4f8KPtWo/8+0H/AH8P+FH1un2f3MPYy8vvRo0VnfatR/59oP8Av4f8KPtWo/8APtB/38P+FH1un2f3MPYy8vvRo0VnfatR/wCfaD/v4f8ACj7VqP8Az7Qf9/D/AIUfW6fZ/cw9jLy+9GjRWd9q1H/n2g/7+H/Cj7VqP/PtB/38P+FH1un2f3MPYy8vvRo0HpWd9r1BeWtImHcLLz/KrVrdR3cJePIIOGVuqn0NXTxFOpLlW/mmiZU5RVyaiiiugzCijFFABRRRQAUUUUAFFFFABRS4pKACiiigAopcUlABRRRQAhIFAIrm/EirdX9tZWe46i5V1cMQLeMNkufrggev4VnaLLIL3S5xK8l1dC4+1oWznae47bTwPrQB21BqrHczNGrG1dSRkqSOKd58v/Pu35igCG809LgFx8snqO9YRBBweorozNJj/j3f8xWRd2U5mzb2zbT1yRwa8jHYXmtOmtep10KtvdkylVnbDDt8wM74BIzx9KhYi3JWWFnkHVcjAqKS5klfc0JHGOMV5XL7NO+51X5n5FtY4J2+RmjY9FxkVXCktgAk+gFRCaRSCsTAjpyKnN/Lt+SDYx+8y4yaHyzV3ow95bAsTvJtC8+/GKL8SLZgFw6AELimNe3DJtKNjvyOaG824sGiSFiwbOMjoaqMY6qJLvuxzSvIBvYnAwKli/495+/A4/GoWjaNtjjDelW4Ld45tkg+RwVbHNKnGcqmvoObSiU6Kne2YZaI+Yvt1FMEMjdI2/KsnTknZo054tXI/rUV7qKwy3NnJboUIwpHBHoasgCAedcfKi8/N/F7Vz88rXFw8rnLMa8vMcVLD0lGLtJ/l5m9Cmqs7vZEdFLRXyR6wUUYoxQMKKKTNAgpaKMUDEpaKKAE7V1HhKXME0XdWB/z+Vcua1vDlyYNUCY4lGPxr2ckrKjj4N9dPvOLHQ56EkjtqKTNGa/WD5YWigciigAooxRQAUUUuKAEooooAKKKKACilxSUAFFFFABRS4pKACiiloASlpKWgAooooAKKKKACiiigDgPjDIw8KWUAP7u41GKOQf3lwxx+YFUVUKgCjAA4Are+J2kvqngyV4WRZrKVbqPe2ASucj8ia8/tPG+kzWiyTytDJjmMqSc+2K+M4mwtes6cqUXJK+2vY9rLakIKSk7G14PHl/F65VPlWTTCzAdyHUZr1XFeVfDT/ieeMNQ8QxFUt4rf7JGhYbmyQSxHbp+teq19HltOdLB04VFZpI87FSUq0nEKoaxzZxofuvNGrD1G4Vfqhq//HvD/wBfEf8A6FW+L/gT9DOj/ERNWJJrM99qc2naJGjPbOEuriX7kR67QOrNg/Tnv0rbrNvdEgurxL23d7S9TH7+HgsP7rDow+tc6sUctrGu6jY6t4ht4ry3jCxIbdJ3beCYRzGB/tfrUc2sXNlfebeLcS2w0K2kuWjnKSIS8gLKO7Hp1FdFq5vbfXrG5gsZLyz+zzJOkQXd5haMxn5iOABJ+dMFlba7dXZv9JvLb7RarbyNMygMgYkAbWODlic1V1YRz93capptnrVzE99aWTWcawf2ozTt5mJi+3a5I48vknHFSzD7bpVhDpmpX7avcQwZWC7fZAMLud1BwOM8HqTW5d+G4l026EE97cTNbyJGs928gyUIHDHHeqtvHqWjTaWYrCWe3Fh5dzHb7dwmGzBOSM8BhRcRm2uo6rd6v4oWxnea6i0qE2avCYgshM+OG68gc+1TaW6f21pg0WTUXyr/ANo/ahIF27Dgnfxv346ds1vaZtuNYub99Mu7OeWGOJ3nK4ZULlQAGPQu351z99Ysb7VDqul319cySsbKe3YYVMDaFORsIPX35oGYel6u8vgjQhqeoySKoikuraeeGGWaIw8bcMCy7iD82Ca29AvNOudSvobS+Ww0+eKNIbb7cgl80FizKFY7QRtHB5wa1JPC7aroWlQapOYru1SNpXgVfnkUDnkdARVGPweY/EUySyTXNhfW5a5dtq/vFKBQCuCAVDAjoad0xGfp139p8HeJ7yLU72ea2S9jj824ZgIwGMTgE912kN3rpIL/AFSTRbK8srJiFT9/bXGFkkGBypBIz14PXPajXvD8Fzo9+bC3C3j6dLaxCM7Q4KEKpHQ89M9KszaP9v0+ztruaVII4wJoEOBKcDhj1x14755qboZZ0rVLbWNOjvbJi0TkjkYIIOCD9CCKktfl1q4VeA0KMR75YZ/ICpYYIraFYoI1jjQYVVGABUVv/wAhyb/r3T/0Jqz/AOXsPX9GXH4Zen6o0e1FFFeocwUlLSUALQKKKACgUUUAAoNJS0AFBoooAKKKSgBaKKSgBaO1FFAFG50ixu7lbi5tYpZlAUOy5IAOQM/jUsFhawXL3EUEaSyfedV5arNFACUtJQTQBHcBjbtsOGwcYrADXbdPN/WujyCKhl3+WfI27u2a48Th/au/M1bsbU6nLpY5+S3nVPMlQgE9TUVW726uJD5U4CleoHeqgr56soKdofiehTba1CijNFYmgGlVmQ5QkH2pM0Zo21CyJxc/d8yNXZejGo/OlyxDEb+tMpa0dSbW5HJEFZlOVJB9jTjPKcbnY46c01VZ2wgJPtSSyxWY33LAsPuxg8n/AAqOfkjzN2QWTdkrso64Wa5icscOgOM9DWZitK7B1R/tFu2X2/NDnlcentWeyOmQ6sCOxFfI5gnOvKrHWL2Z6uGtGmovdCUUZorzjqDNOijeaVY4huZugqe0gRlkuJ+YosZA/iPYVYk1OxFuJ1lhjl8soEHHzE8n8q9Chg1OPPUlZb26tdTmqVnF8sVchFkkKlr6TYOiohBY0jW8EsLvaO5MYyyOO3qKz2uYt+15kDA4wXGa0NNOZpU/vwsAf1/pTpunOap8lk+vX7xSUox5nLVfcVaWm/WlzXndTqA0GijNAAa3/DV3awrIs5RJM5DN3FYFJXbgcZLB11Wir27mFeiq0HBux1Gta8oTyLB9zt9517fStbS0lXTYRPnft5zXK+HrQXWpguMrGNx+vau2A4Fff5NUr4yUsbVej0S6HgYyMKKVGPTVscOlLSDpS19KeeFFFFABRSVANQs2kkRbqEvEMuokGUHv6UAWKBUMF1b3UfmW00cyZxujYMPzFTUAAoNJRQAtBopKAFNFFFABRRSUALRRRQAUUUUAFFFFABRRRQAVkeKPEMHhjw/Pqdwhk8vCpGOrseAK168++L2X0XSIUPzSanH8ufvABv8A61RUlyQcuxUI80kjmbqPWvF0q3Pia5aG16x2EJwoH+161pR6bZRIiR2sSqgwoCDirI6UtfkOMzLE4yfNUl6JbI+tpYenSjaKMS40F7e8/tDw/cvpt8DndGcI/sRXZ+CPGVzrN1PpGuQLBqlsgclPuzJ/eH6fnWPVfw9BNF8Vbe5jt5HimsXikkA+VDnIyf8AgP619Hw9mdeVdYapK8WtL7nn5hhoezdSK1PU6y9QuY7l4ra2zK6zKzleigHJyas6jaSXlr5cMxiOc8dG9j7VUtpkt2W2mhFtJ0UD7rfQ19bi6kr+yatF9e/l5fP5Hk0oq3OtX2LtFFFIkbIzJE7Iu9gpIUHG4+lc/wCGPEV7rWo6vZ6lp6WMunzJH5aybyQyBxk+uGrfmmjt4JJp3WOKNS7uxwFAGSTXA+GPE+h/8Jv4sk/tW02TTwyRt5o+dUtl3EeoG05+lUldMRu6v4kv9M8VaVpo01TZX8/k/a2l53eWz4C4/wBnqTRe+Ir2416XR/DtpDPLa7Td3Nw5EUJYZCADlmxzjtWF4w8TaIfEHhVhqtpiK/8AOkPmj5Ua3fax9juH5isOOz8MCXxVc+IbpVvJbuW5tXaYgtE6honh55PQcc5GKpR0Fc7jUfEl/YeMtK0g6aos7+R4xdtLySsTScL/AMBxyawb7xn4li0bXNYtbbSzZ6VdzwCOQSeY4jkK5yGxk1nS+JLaO78A/wBvajBFqFsWa+WWQBoma1YZcdiSR+JrQHhvStX8QavoaX+o/Y7qMajNHFcYhk892JAA7ZUn8adktwN238VKPHt5od7NDEotreS1Q8NI779w9/uiuem8a+KIvB8viYRaSbOO5MX2dkdXI+0eT97dgeucV09jok0PjzVNVmjjNvPaW0ULHlgyGTd9PvCuS1zwrajXbfw3pCyyfaEe/eK8u5TbRqJQcCNWG4lznHQUly3DU3/CXi278ReI9YtLiG3hhs4bd41gnWbl9+csOP4RxXX1w/gm1Fr4u8RR3NvBb36JbRzLaDEDLtYoyjqCdxyCT0FdxUytfQaCq8H/ACHJv+vdP/QmqxVe3/5Dk3/Xun/oTVl/y8h6/oy18MvT9TQooor1TmCijFFABRRRQAUUUUAFFFFABRS4pKACiiigAopcUlAGTPrTW+pR2kto6iYssMhYYdlXd07DA60uh64utWslwkPlRpK0atvDB8cEgjtmqMdhqZ8QXN9dQQTqA0dqTLgRpj+7jqT1P+FTaTo7xT3sl1BDBFc7QLaI5VcDBPbk/wBBQBu5HrSZHrUIsoAP9Uv5VWlNrHOIhBvfuFXpUynGCvIaTexf3L/eFIZE/vL+dRi1g2/6pfyqjqYECBYoVww+9jpU1KipwcmOMXJ2JrvUYLeM4kUvjgA1iLeOjblnwc5+9UZtuMmLj3Wm+VH2UflXzuIxU6sk9j0KdKMVbcmnulncOzruxg81F5if31/Ojyk/ur+VHlJ/dX8q5ZPmd2bJWVkI0o8l3jdCVB5J4B964q08ezTuVeG3c/b0tB5O4jaX27yT2IOQehrtGgieF4mjUxyAhlxwQetcxFol8xS3vNOs5Io7kOk6TbGEavuQYC9hjjNXT5LPmJlzdCtL8QGivJLU6a5lW78gHJwV8zbu6enPWuzByAfX0rjtS+H9tJdG9sJNtxvXEbgeXt3gsMY9M12MaBI1VQAFGAB2oq+zaXIEebqLSgbmA9TRSh1gje4fkRc49T2rFWWr2Ld+hW1G9W2R7a3bMh4kf09hWMSWOSc+9PdzLIztyWJJptfHYzFTxNRye3ReR61GkqcbdRASpypIPqKtx6ncKRvfzE7q3ORVWg9K56dapTfuOxcqcZbos3Nuhj+0WvMR+8vdD6VVq9p6PHDLcScW5UqQf4jjgVRIrfEQTUaiVnLp+vozOnKzcW72Lun5jWWZpGjjXAOBncT2xXlOuxNaaDo9rHYS2V1GG+1TttxMwHbBP517ClvFb2pSW4V4Jk38j+LHashNNtdSl23sSyRRqXO4ZwK9ihWnhFCi4p3++/8AlqcklGq5TvscXrKXN5PHYJo1utzdKZEuPtHOEZCf4epzXdiS7sbFLdhFHK43SBTuKggcZp7W2l3PlXLIsEsJZV4LNtOOn5VFdzi4umkAwDwB7VzVa6pUfctfpZt77+nRfeaxj7SeuxW/ef7P5Uv7z/Z/Kn0Yrx+Y7LEf7z/Z/Kj95/s/lVq1s5r2by4Fye5PQVpv4bl8v5J0L/3SMV34fAYvEwdSlTujnqV6VN8s5amF+8/2fyo/ef7P5VPNBJbTGOZSrDsaZ2rhlzQlyyWqN1aSumb3hRJ99w6eWBgDkGum/wBK9Yv++T/jWZ4atzDpYcjmRi1bVfq2TUnSwNNPe1/vPlsXJSryaIf9K9Yv++T/AI0n+lesX/fJ/wAanor1zlIf9K9Yv++T/jSf6V6xf98n/Gp8UUAVybsSpnyimfmwDn8K4+aGHxHqF80MkdokEbwxKV+aQ71Z3YdduVAx6Z9a7imiJAxIRQT1IFAHOeE0uWk1G7niWGG4nHkoiFQQqKpbB9SDXS0YxRQAUUUUAFFLikoAKKKKACilxSUAFFFLQAlLSUtABRRRQAUUUUAFeefFS0v7i40GbT7Ke7FrcPJIsK5wPlx/WvQ6KipBVIOEtmVGTjJSR47/AGnq3/Qtaj/3xR/aerf9C1qP/fFexUV4H+rmX/yv72d/9o1zx3+09W/6FrUf++K2/Ces6imuxwz6DfQRzjYZXXCp3ya9HorehkeCw9VVaad15sipjq1SLhLZhVbULY3VoUQgSKQ8ZPZgcirNFexOCqRcJbM44ycWmjKXUY1G25V4JB1VlJ/IjrTv7TtP+ev/AI4f8K0ioPUA/UUbF/uj8q4/q1VbTX3f8E19pDt+P/AMxtRs2Uq0mVIwQUPP6VRvIdHu7Ge2Cxw+fG0ZdIMMAwwccehrodi/3R+VGxf7o/Kj6vW/mX3P/MOen2f3/wDAOetINDs7KC1hgh8qCNYk3Q5IVRgDJHoKnkk0mZo2lihdo/uFoc7fpxW1sX+6Pyo2L/dH5UfV638y+5/5hz0+z+//AIBjPLpUrl5I4XY9WaHJP6Vm6novhjWLpbnUbOGaZYxGHMbA7QSQOB05NdXsX+6Pyo2L/dH5UfV638y+5/5hz0+z+/8A4BzGk6Z4d0OaSXS4Et3kXaxVW5FLrFhoeuGF7/d5sGfKmiLo6Z6gMOcGum2L/dH5UbF/uj8qPYVt+dfc/wDMOen2f3/8A5zR7XRdChkj04FPNbfK7BmeRvVmPJrR/tO0/wCev/jh/wAK0ti/3R+VGxf7o/Kj6vW/mX3P/MOen2f3/wDAMw6na4+V2Y9gqMSf0qXT4ZGmmu50KNKAqIeqqOmffk1eCqOij8qWrp4eSmpTle21lb9WKVRWaitw7UUUV2GIUlLSUALQKKKACgUUUAAoNJS0AFBoooAKKKSgBaKKSgAKijFLRQAU0Rru3bRn1p1FADelVrq8ggIEhyfQCpp5VhiZ26CueulLP5pfeHPB/pXDi8Q6Mfd1ZtSpqb1L0urKQBHH353elZ9w6STs0a7VPaoqmgRGDtJnagzgd68OdapX92TO6NONPVENKKnEsTsEeJUU8Bh1FNFu3msjEAJ1Y+lZ+zb+HUvmS3IsUEVJLF5YVlbcjDg4qOs5RcXZlJpq6DFAooqRhUOpTfZrERqu43A5bsBn+dTUsiGaxuI9u8BMqMdD7VFWMp0pxi7OzGmlJN7HPCigcUV8Ke2FB6UUHpQBLBdy26lUOUPVSMg1N/alwMBdqr/dCjFU6K6I4mtCPLGTsZSowbu0ar3dgbcq6O24hxGOApxzz6VFBfW6MY1tVSOQbZGyScVn4ox710PMKrkpWWnkZLDQStr95cfSp/NbYAUz8rlgAagktJ4m2vE34DINMLMQAXJA6DNSx3dxEu2Od1HoDWblhpPZr7n+GhaVZdUySKy2oZbwmGMdBj5m+gpywWcxxFcNEc9JBnP5VVkd5W3SOWPqTVrS7mC1uw1xEHU/xEfd961oSoSqRp8qUXu3/wAAipGoouV9eyOisrGPTbdlDFyfmZsVkaR4zsNY1CO0t45kaZGeJn2kMFxngEkdR1ArVu2i1HR7j7Ozy7o2GIztbIHQeleT+F9A1JvE8TyW+pWcU5YzNI/KfLxghRg5APWv1fCwo06MY0Lcq2sfL1JTc257nol1reizSJcXG5tt4dPyxwokBPXnpwa1JdGsrogLH5R7MlcJp2iWEv2KE6fe2U4u1muImjZ45HXcNxJOMnOS3tXoqi4WYbIFaPI2nfjPrxis6mAwtS/NTTv5DjWqx2kzQgiWCJY0GFUYFSioIZneZo5IwhVQww2c5z/hU9dcYqKsjNu4vaiiiqEFFFFABRSUtABQKSloABQaSg0ALQaKSgBTRRRQAUUUlAC0UUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABQelFB6UAJRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRSZFAC0UmRnFLQAUUUUAFGaKhufN8hvIAL9s1MnZXGtXYz9XuSAIF7jLVSQCe12A4dDkZOMg1FMZDM3nZ3981H3r5mtXc6rk0elCmlBJFj7Mo5eZMDrg5NNkkQRiOHOM5YnuaiALHCjJPYVZ+wSGEMPvf3D1qIqU0/ZxHJqL95lWpWuHePYcY7nHJp8dlM5+YbB6tTLiBrdsHkHoR3pezrQg5NNIrmhJ2JJcPaRsflI+UDPX3qJYJXXcqEj1pUuHVAmFYDoGGcUjzyOwJYjHTHGKJSpy1YkpLRDDx1p0cbSk7B06k9BUn2nPMkaMw/iIpJLhpE2gBBnJCjqanlprW47y7CPA8Y3HDL6qcioZbkWNr52CzuSqDsPepYZWicYPynqD6Vm61LN54idFSIEmPb3HrXJjKyoUHVp3T29PM0pQc6ihLYzs5OT1NFJRXw71Z7YtB6UlBoGYGreJG0m+EM1qskTKzbkmG5VA6kHGBnA696fpnii0v8Ccx2rNDFMBJIBu3rnA+nSqeu6RM99I9tbSPDPCc/Z2CsZc8Fs9V6VHc6Nfq1iFhLxQ2ixutvII8SD8Ole1GlhJ0o3sm+tzjcqikzc0TVRrGmJdrE0QYkbW9j29a0azPD1nPY6DbW92AJkXDAHOOfWtOvKr8iqyUNr6HTC7irhRSUVkWFFFGaAJ7S7ezuUljJ+U8jPBrq7S+t70Zgf5sZKEciuVs7c3Fwu4Hywcu3YDvVs6nHaqw02Py3c/MTzx7V9Pk+PqYGLc37j6dfl+B5WMoRrStFe9+HzOnHJxV9FCqAB0rP02O4ltI2vBiUdfetKv0alUVWmppWueBKPK2isZVjvnMh2gxrg/iak+1Q/89BUu0HqAaTavoPyrQkZ9qh/56D8qT7VD/wA9BUu1fQflRtX0H5UAR/aof+eg/Kk+1Q/89BUu1fQflRtX0H5UARfaof8AnoKX7VD/AM9B+VSbV9B+VG1fQflQBF9qh/56Cl+1Q/8APQflUm1fQflRtX0H5UAQG6jlYwwyDzSpK8frXNnUtV/4Qm5vvtUf2y3llBk8oYYJIV6fQV1Dx5RgnyMRgMByKxR4axos+mm+nMU7s7Nhc/MSSOnqaANuElo1J7gGn1FbRNDbpG7tIVAG9up96loAKKKKACiiigAooooAKWkpaACiiigAooooAKCQOpxRXmfxV8yXXvDdqJpYop2nDiNyucBPSs6tRUqcqktkr/cXCLnJRXU9L3D1H50bh6j868W/sKw+1fZvt93523fs+0tnHr1obQrBLlLd7+7E0gJVDctkgfjXz3+smD/ll93/AAT0P7Nq90e1Ag9DmivFN1z4R1bTb/T725aKW6SCeCWQsrqx969qByK9nB4yljaXtaW3mcVajKjLlkLRRVTUp3gsj5JxJIyxqfQk4zXTUmqcHN9DOMXKSiiWW8toW2zTxo3ozAGmf2lZf8/UP/fYqvDZQQpgRhj3Zhkk+pqTyIf+eSf98iuH29d9EvvNuSn5kn9pWX/P1D/32KP7Ssv+fqH/AL7FUZ7vT7e8htJPL+0TfciVctj1IHQe5rCk8ZaVFaXMjQL58F61n5GOWIm8rdnGPen7XEPt+P8AmHLT8zq/7Ssv+fqH/vsUf2lZf8/UP/fYrno9ZFxqGqWNnZ2013Y3CRLA0wRnQxRyF+h6F8dO1UbjxfBaWN013pyw3dveJaeSuZFYkx87wuBxIOuOlP2uI8vx/wAxctPzOv8A7Ssv+fqH/vsUf2lZf8/UP/fYrnl1W7udQvrfTtEhnSymELySXIj3MUV+BtPGHFFnq11f2bS22iRNLFcSwTRm4ACMjbeDt5zz6Uva4jy/H/MOWn5nQ/2lZf8AP1D/AN9ij+0rL/n6h/77Fc3Z+IIrw6XiwWI31xPAykg7DFuBIOOQSn61amk1ZZ3WDRrOSMMQjtc4LDscbeKPa4jy/H/MOWn5m1/aVl/z9Q/99ij+0rL/AJ+of++xWFYa5Y3Hh2z1a9gS0jusA5G5Yyc9TjgcdTxWwsUDqGVI2UjIIA5o9tX8vx/zHy0/MmXULNmAW6hJPQbxVjtVFrWB1KtChB/2aZp5aC6ns8lo0VXjyc4ByMfmKqniJ86jUS17ClTi4tx6GhRS9qK7jASilpKACiloFACUUtAoASilFBoASiloNACVi6rNdQa7o4huCsE8zxSw7QQ/7tmBz16rW3WbfaNFqF5bXMssyvavvjCSFRnBHI78EigDM028u7fV5oNWacSytM8CnBRo1fjGO+0rW3HfROgYB8EZ5Q1BDo8MWoi8Z5ZZVVlTzJCwQMQTgHp0FaFAEH2uP0f/AL4NH2uPHR/++DVikY7UJPYUAVP7Qg8zy9zbsdNhqFNWjabyzFKDnGdhxVpY4pmWYAE9mqUrleOD61j771uVojA1Efv2mUOytyfkPFUPtCf7X/fJrob27SCIozB5CMYFYdeDjY041Pd3e56FBycdSxpc0b3DqoYuEyPlPrXKfEV7hJrby7aTZIjRidJCHDnldqjk4wSa6RWZCChKn1FMubK31VIxezyxSQSeZBLGxVkOCDg+4OK6cFi4QSpNW8zKvSk3zI4Oa5u7zwlcNeXEsVvPdXMouFRna1j3NkgjHzD36eldb4HUSaObS4SRzG4KTmNk81SPvEN3+nFaMOn6XaaLNpaLJNBMJBLuYkyF8liT1ySTVsXccVqsVshTaoUZP3R0ruq4rD2abuYRpVLpoquAJGC9AeKSgUGvmW7u56i2Cig0UAFVtVgkmtYJIxuEYIbHbmrNOjcxuD27j1FZ1KSrQdKTsmNScJKS6HOUtWdRtjbXbYH7tjuQjoRVUV8TVpSpTcJdD2ISU4qSFopKVlKttYEH0IqLO1yrhUUShQ2P7xP61LTI+h/3jVJ+6xPcfTo42mmSKMZZzgCmgEtgDJrQsrO6t3S+aFhHEwJyO1bYehKtUSs7dfTqRUqKEd9ehUurZrO4MMhBZeuKh7V1t5oMOpSm6jlKmQZ45BrBttMlbU/IlidkVsOQMcV6WNyivQrJRj7snaPX0OajjKc4Xb1W5Vgtpbnd5SghepJAxTLu8stAkt5dVlX965SNEG/LbSecduK0L5ltPMtYYFUSAHcM5I7VzXizQLyewViFDRo0sSNGDubBxg9vT8aVKhSoVVGSb5d3rZPp0/4ASnOpHR2T27k0ni+3urG9uBM32eyOJcJt7A9Pxqxpd/b3fiBNPwXlCJKVJxlWz/hWFD4MnttJhGo2z3UVy8U01pCu3eCQG3Y64HOP9muy8G+G9P0/XLm9tUu/+PdIYxdxEMgBY8EnnqPyr2sPkssTU55N2T6/L/g+RyVMXGnHlXU7pBgAU6kWnGvvEeEJRS0lMAope1FACUUtFACUUtFACUUtAoASilFBoASiloNACUUpooASilooASilooASloooAKKKKACiiigArzT4n/8AI1+FP9+f+SV6XXmnxP8A+Rr8Kf78/wDJK48d/ulX/C/yN8P/ABo+pzWoW+oweLk1GytPtERtxC3zYx82f6Uut2+oL4isdRsLT7QIYmVl3Y5NPutZ+xeMltbq5EVo1ruCt0L7uK6DrX5jOvVoeznKKfu266p6a+nkfSqEZ8yT6lQaloUep2cPiCa3ULIsnly87WA4P616nBPFc26TW8iyROMo6nIIrzj4YaFpmr+G7rUNTtIru5ubqRZHlXccA8D2rR+GDNFa65p6MTbWOpyxQKTnYnHFfomVYBYGh7PmvfX/AIY+fxdb28+a2x3VUNX/AOPeH/r4j/8AQqv1Q1f/AI94f+viP/0Ku3F/wJehhR/iImooormKOc0rT7rQtTkSSH7dFdyFjff8tVJ5Akz1HYEe3HeqC6FaXOt6jpJvbhYUlS/eLCBS0krSYBxn7yHvXXTzw2sDTXMqQxIMs8jBVUe5NYV7L4V1HTTr961nPaKPL+2HkYD7cZH+0SKpNiMzUvC1zd3WpXEdvb3HnarDdLG0m3zIlt44ypbHHzKTj2qpr+lzW3huXbp0ek26XEMjR21zvErmaMfMMDPArpfD13oEyzx+HWhIUhpREpHXp1+lGq+INAt47iDVbyDbBJGs0bAttZuUBAB5O3I+lO7uBy893Evia9ns7jVbe0uYBM4hhkAa4ACen91Vq34aN+/hu3sdOaWO+nUzX93dIwaKR+XKgjDHJOOwrbg8Y6Fc297LbXwkFjbm4uF8tlKRgE7sED+6fyq3da/pllYW95eXSwwXABiZgfmyM9vai72sBz2t6FZrc+GdJRcQRyTKhPJBELYbnqc881Nqcet2WqaRaR+IJil9O8Tk2sWVAiZ+Pl9VFXpPEvh+ezGqC6iuIrW4WESKpJjkchQPbO4fnUmsa9oumX0CaqzCdB5sRFtJJtzlcgqpAPUUagQ2um3Hhzw9Fp1v5urhQsEMciogVQP4iB045PJq14d0iXR7GSKefzGllMgjQYjhyB8iDsvGfxNGleKNI1u7ltdNujJPEgkeNonQhSSAcMBxkGtapd+oBVe3/wCQ5N/17p/6E1WKrwf8hyb/AK90/wDQmrP/AJeQ9f0Za+GXp+po0UlFeqcwtJRRQAtFJRQAtFJRQAUtJRQAtFJRQAtJRRQAtJRRQAUMMrigmo97+bgr8mPvZpN2AAoiQBV46YFPyM4zzRmo5CUZWVNxJwT6Ck/dQGNqygXxI7gE1Sq1qTK96xVs+vtVSvlsS060rdz1qXwIWkrI8U+YfDtwsMW92XAYybBH/tk+g61xh0/XbaOC2nn+zmPT5yZAxKFlMZ3luvIB+nNTCkpq9xylZnpVLXC2+t3Gt+I9FCTC1VfMfEYciZdo4+ZQK7qonBwsmOMrhRRSVBQtFFJQAtJT4VV5kWQ4UnBIrUXR1EoYuSnXFdFLDTrK8TKVSMNzJnG/TJ/NGUVcqT2ORWLY2j3t0kCZBbqfQetd89tC8JieNSndcVnpPGL5Y7OBB2ZsdqjG5LGrWpzqT0282FHGOMJKK1INL0T7JIwuo4pQOUfvXOaoGGqXAfrvPSu9LKrAZ5PQVzmseHnd5Lm0YuzHcUP9KnOMqtg408LG/K7vv/wQwmK/fOVV7nNUyPof94/zrqNN8NJJa7r4MsjdFB6VE+kQWeiXHmHMvmMVb6E4FfOxyLF+xdSSSVrnoPHUublWruWNI0W1kht7reXf7xGeM+ldAUVlKsoKkYIrD8KxstjI5zhn4rX+225kKCaPf/d3V9vlUaEMHCSio8y+88XFObrNN3sB8mytgANsa8ACpVCnkDr7Vy2sateR3yQtGqeW+5QOd/pXTWzu9vG8i7XZQSPStsJjqWIrTowVuS3QmrRlTgpS6hJZW0sgkkhRnHQkVSu9IS8v4552JjjHEfbNalJiuyrhaNVWlFb3+a7mUak4u6ZVWJBqG4KMiIDP4mrRUVWeRI78l2CjyxjJ9zUv2qD/AJ6r+ddJmSjApah+1Qf89V/Oj7VB/wA9V/OgCakqL7VB/wA9V/Oj7VB/z1X86AJqKh+1Qf8APVfzo+1Qf89V/OgCbNIGHrWdqOs21lYTXG9ZDGQoQHlnP3V/HNczZ+IdXvbDiSCO4t7aW4mKqCrlXZVT2HynJ69KAO3zS1Wsbj7XYwXGMeaivj0yKsUAFLSUUAFFFFAC0lFFAC0UlFAC0lFFABS0lFAC0UlLQAUUUUAFFFFABXmnxP8A+Rr8Kf78/wDJK9LrzX4o/L4m8LStxGssylj0BIXArjx3+61P8L/I3w/8WPqZFx/Zd1ry2dzapLdiHzQ7JnCg461LqmrxaZ5USxSXF1NxDbxDLP8A/WqhqWh3txr6alYXyW5EQiYFMkjOTWv4Ujab4sOzqHS30w8n+Fi4/wDr1+eZfgqONxFKk53XLdrXRrp8/I+gr1p0acpW1uN8N+CfGEFpdTW2rJoiXjmT7IE8zbnv14NWfC+pzfD/ABoviexMMc85ZdUjJaOZmP8AEexr03I9a5j4jQC4+HurqEDssO8e2CCT+Wa/TIwUIpLofNuTk9epuahePaWglhiM2SBx0Uep9qqQQC6KXNxP9oYHKbfuL9BUvh+Zrrw1ps0nLSWsbN9Sopl/aizU3lplCrAyovRlzyceuK4cVTnf2j1iunbz8/n8jalJW5Fo+5bopFYOoZSCpGQR3pakkiubWC8t2guolmifG5HGQcHNedaFoc+u/CI6bYPFDJ/aUzp5i5T93es+CB2+XFegala3F5YtDaXsljKxGJ41VivPowxXNaV4Kv8ARdMaysPE96qeY0qboYjtZmLN/DyCWPFXF2QjL1Lxzc+FJLjTdctLGHUpYkaxktyVimLMV+bPK7TyeelcjDDanxDqMWk3kOq3Mdxp19M3nAG5KGRpWBPHf8BXqOmeFVt9Tm1PWLttVv5Y1hEk0ahY0BJ2qoGBycmmXPg20ebXbi2Iim1ezFt/qxthwjKCMc87+fpVKSQrM5jVrbUby18V65dacthazeHHt4l80OXKrK2eBjGHFbulXNp4c8EWU2u373YeJGiWVAzklBiNFA5ptnp/jCz0eDTidEmhigEB8yOU71C45+buK6WC1DWtt9thgM8Kj7i/KjY5256Ck2M821rS7y1+HupX9xEtpe6tq9tdCAjIgBmiWNTjqQqjPvmukl8UanpGmzyaxoF1cfY0dprqBUWNlXJ3AFs4wM1reKtFk1/Q/sMEqxN9ogm3MMjEcquR+IWrWt2D6p4d1DTo3CPd2ksCsRwpZCuf1o5k9wschol9qF/45k1w+H7yGyvLGC3hkJTCgM7lm56fOK76qmlWjWGj2lo7BmghSMsOhIAFW6lu7AKrQf8AIbm/690/9CarNVrI+bq1zKnKIixZ9WBJP8xWf/L2C8/0Za+GXp+qNGil7UV6pzCUUtJQAUUtAoASiloFACUUooNACUVV1We4ttLuJrMI08cZZBJnbkDPOKwJvFjRLpsbmOOW5tlup3ZWZYk4z0988npigDqaKgW9gLFTKu4AE/jTvtUP/PRaAJaKj+1Q/wDPRaT7VD/z0WgCQ1DcBmjwF3L3wcGnG5h/56LUFzdhISbdlZ89DUT+FjWrLKHKjtx3qKK5juN4TnYcGmG5R7c5dVcr69DWCl6bSRgsgVuhrkr4h0XHTQ2p0+e/ckupRJMdsYTBxx3qCmG4jJyXBJo+0Rf3xXzs25ycj0YpJWGXdrFfWUtrcLuimQo49QRg1nR6Nb2n7t7i9nSZDCY5Z2dcEc8H2rU8+L++KY8iPLFsOfnzx9DRFyWgOxnWXhqzsrq3uElupGtlKwrLOzqgIwcA+wrXpRQATwoJPtScpTeo7JCUoRmBKqSB1wOlaFvpTSRFpSUY9BVzZFptmcjdk8+9dlPBTkuaeiMJV0tI6sx44TJjkKD03d62rXT44oQHVXbqTioL2eFFgKBWw2cD0q9bzxzxBo2z6j0rvwtCjCbje7OerUnJX6FW+s0a2JiQKy8jA61QXVJ1iCcEjjJFaWpXD29uDGOWOMntUcFhFJZgSL8z/MSKdanOVZqi7O2v6BCSULz1DTGlliaSZy2eBmqJL2OpHYN3oPUGtmCFLeIRR9BURs/9O+0bj0wVrSeHnKnBdV1/MmNRKTfRlZ9RhE6t5bb8YPt7VoooCADpis2402WS781WBUtkj0FagGBWuH9rzS9oRU5bLlDFVoVVkk3KCPMbqPerdVbcfLJ/10b+ddbV1YyILG9trpZI7RSojOCMYrnX8P3CLPPNKIwmSvPWupt7OK1LmFNvmNub61X1m1+06dIu5lKjcMHrXhYzAfWMOpV1eUE7JaI7aNfkqWhonY46Nru6mEoDTNCB74ArtdNMj2Uck7bncbjx09qo+HbRI7FZRHteQcnPWthECKAowKxyTL50Kft5ybc1e39eReMrqpLkS0Q+iilr6Q88aVB6gH6ik8tP7q/lTzQaAGeWn91fyo8tP7q/lT6DQAzy0/ur+VHlp/dX8qfSUAN8tP7q/lR5af3V/Kn9qKAKtzp9rdx+XcW8cibg2GXuOhrGXwbYQ24htHlt12yI5jON6OclTXR0UARQxLBEkUYwiAKo9AKkpaKAEopaBQAlFKKDQAlFLQaAEopTRQAlFLRQAlFLRQAlLRRQAUUUUAFFFFAARkYrhrz4VaTqB/0u+1CUBtwDTkhT7V3NFK1xptbHC/8ACqNJH/L/AKh/3/NEXwp0mCZpob/UUkYYZ1nIJHpXdUVEaVOLvFJFOpJ7s4z/AIVtZ/8AQX1T/wACTTZfhjp80TRy6pqbo42srXBII9K7WirsieZlXTLCPS9Lt7GBmaO3jEaFzk4AwKtEAjB5FFFMRQOkxKx8iWaBT/Cj4H5Uf2Wf+fy4/wC+6uvLGhw7qp9zim/aIf8AntH/AN9CuR4fDp7I29pV7lT+yz/z+XH/AH3R/ZZ/5/Lj/vurf2iH/ntH/wB9Cj7RD/z2j/76FL6vhu34j9pVKn9ln/n8uP8Avuj+yz/z+XH/AH3Vv7RD/wA9o/8AvoUfaIf+e0f/AH0KPq+G7fiHtKpU/ss/8/lx/wB90f2Wf+fy4/77q39oh/57R/8AfQo+0Q/89o/++hR9Xw3b8Q9pVKn9ln/n8uP++6P7LP8Az+XH/fdW/tEP/PaP/voUfaIf+e0f/fQo+r4bt+Ie0qlT+yz/AM/lx/33R/ZZ/wCfy4/77q39oh/57R/99Cj7RD/z2j/76FH1fDdvxD2lUqf2UG4e6uGXuN/WrcMMdvCI4VCovQClE8THCyoT6BhTz0rWnRpQd4IiU5vSQUUlFbmYtJRRQAtFJRQAtFJRQAUtJRQBX1C3e7sJbeKXymkUrvxnGfasNvCrNYxQfb3V1tDZySCNcyRensRzz710lFAEMNvHDCkagYVQoyPSpdi/3R+VLRQAmxP7o/KjYn90flS0UAJsT+6PyoKLj7o/KlNZuoX5t28qMfOR1PasqtWNKPNIqMXJ2RYuZooUPKBhzg96py29tqKb4cCTFZbM0jbnJZjWhpVu4uDI6lQoxz3ryY4l4mp7NxvFnU6SpRvfUqXFg9qAXVSD3FV9q+g/Kt7VYme0GwZwcnFYVcWLoqjU5Y7G9GbnG7E2r6D8qsNaqtik44JbBpsdtNL9yNjzitO9t2XTo44lJ2kZxTo0JShKTXQJ1EpJIyM8Vt6XCFtQ7qNxJwcdqy1sbhgCIj+NdCgCRqDxxiuvL6DU3Oa2MsRNONkMnnS3j3SHArOvb+KWz2IdzN+lXLk2zny7gjjnBqqbO2uoGa2+UjgGuzEOrK8INbbdTCnyqzkZIBJAFXoba7tZ0KgkMRnHSooLCaV8bdoU4JNaRne0uI4pX3o4xkjkGvNw1H7dS67M6alS+kS48ayptkAYehp2MLhePSloPTivoLLc4DPsYZo7qYzZ9j2NaNQQu5ZlkXBB6joamqKMFCNkOTbd2OopKK1JFqva/wDLT/ro386nqvaniT/ro386ALNNYA9elLmjNADVULgAYHtT6aSBWLqGuPZ+ILLTkijf7Vklmk2lFHU47844pJWQG3S1i6JrT6t5pZIowrEBBLlxgkcjt0rZFMApaSigBaKSigBaSiigBaKSigBaKSigApaSigApaSigAooooAWkoooAWikooAWkoooAKWkooAWikpaACiiigAooooAKKKKAMDxr4jPhfw1LfRRiW4ZhFAh/idun9TXm/wDZ+vX2LnUPEd6tw/JWFsKvsBXT/GFSfDOmuB8sepxM57KMMMn86pjoK+R4izDE4R040ZWvf9D18voU6qk5q5c8A+JtUm1u78O69MLqeGITwXGOXTIBz78ivQK8q8I/8lgm/wCwUf8A0NK9Vr6DL606+Ep1Z7tI4MTBQqyjHYKq6jcPbWZaL/WMwRM9iTjNWqoav/x7w/8AXxH/AOhVtiZONGTW9jOkk5pMij063C5lXzXP3nfkk1HdR6ZZQ+bdrDEhIUFu5PQD1PtV+uYuLSew8SSalqMDajayMBDIvLWY4GNnQjPO4c/WuGNGn/KjZ1J9yW51rQbO41GC6NvDJYKGZZHVS4Kb/lBPPBxSw6nYT3Rgh08yMbGO9UDGWDlgFA9fl/WqOq6JHc+Jrm1S98qTWLZpSptkfasapG2GPT76/rVbWfCt1Jc3LWMRkeDR4ba0mLbW8xXcnB7HaR+dV7Gl/KvuJ9pPuX38QaVb/b1v9PeylsrdJ2ScAbw2/aAQTyTG1OXWbOW4igtdHmuZXs4rthEFwiyFgoJJHPyNWXfaXMukak1rp09i8tqRPczOspKIjkLhge7Hp61SXUI7a+0Saw1WS0eexjtLwrbeYEWNXZTypwdzkU/YUv5UHtJ9zpYNX0xodTkvLCSz/syPzbgSqMhdpbIweeBSWuoxNe2sGoaMbIXoP2Z2cNuIXdtYD7pwCcc9DWZ4VNtq6awmq3j3k+pSSwSRvB5YaBGaNTwAPmTB/GmwJeDUpLi2iuNWg0d3hiNxcKhVwuG2jHzEAkZJ9aPY0v5UHtJ9y3aeJNP1DRdNvLDS2kn1IqsNu/yYJjLnLEdAAeRVy2v42ubuzvNG8u9tokm8mFxJ5iMSAQeOcqeK5M6VdWXgzQ4tIWTUbWRIHRvKJeLCAGQHcNvBOAPeptEu9XsdZvbeC0Z7p0WSP7RERJdINoOXLHAVn/Kj2FLpFB7Sfc3odesZ9M1G9XRrmOKwjnaQyoqgtFkMgwTzkGtdW0vFqJUiie6XdEj8buASB781zFzpep6F4U1+2uZBeW93ZXV08w4MUzIxdcf3SSSPTpWnLawQeG7PT0sJdSkuI8xpIxIQ4GWLn7oGR70nRpfyr7h+0n3N5tOtGXHkKPoKdp7vHPNZyMXEYDxs3Xac8fgQaraJZ3dhpENvqF0bqdM7pD6ZyBzycDjJ54qxb/8AIcm/690/9CapjGNOrBwVru34MfM5QakaFFFFescoUUUUAFFFFABRRRQAUUUUAFFLSUAFFFFABRS0lAAaqXdjHdMGbhhxkVboxUThGpHlktBxk4u6KVvp0MB3Y3N6mrgApSKKIU4U1aKsEpOWrIbkOYWEWNx45rCtIC98qMNwB+auiIzTI7dIvuKBXNXw3tZxl2NYVORNFRbkRXphZ+CcBcdDVlkZmbBIyuBz0qO5sI7hg+SrjowqwikKATk+taQjO7jLboRJx0aIYzIrCOQFuPv9qlkjWVdrdKfiitlGysySvLZwzSK8i5K8VJHEkS7Y1AHoKlpKShFPmS1C7tYTb6VWurNLpRuJUr0I7VaNGKcoRkuVgm07ojjUoiqTkgdfWpKMUtNKysIbiilopgFFFFABUBtISxO3BJycGp6KAIPskXofzo+yReh/OrFFAFZ7GBwAwPBB+8e3NZuraVdalcRpmBLdJY5N+wmQbGDYB9yPyraooAxrXSrka2NQujAuyFolWBCN2SDlvy/nWzRS0AJRRRQAUUUUAFJS0UAFFFFABRRRQAUUUtACUUUUAFFFFABRS0lABRRRQAUUtJQAUUUtACUtJS0AFFFFABRRRQAUUVxnxS1W403wisNlK0M1/cJaiReqhsk/oKUmoq7Gld2KPxC8VaHcaVc6AqNqd5OuFhtzny2HQk9sGuJsbXxbBYxRefbH5cZk5ZP8a3NL0a00qALbR/OfvyNyzH1JrQr83zPPVipKMKa5V/Mrv/gH0eGwXsldyd/IxPBGtQeEtbu28WRSJdXrBRqH3owv932FexW9zDd26T2sqSxOMq6HINeZ3FvDdQtFcRrIjdVYZpngCSXRvHl1odu7nT57T7SkbnPlsGA+X86+gyXOo4trDzjaSXTbT8jgxuDdNe0Tuj1OqGr/APHvD/18R/8AoVS6heiwt/NMbSZbaAOg9yewqokMl2yT3kocKQyRxn5Aex969rFVU06MdZP8DipQatUexcoooqBFDUdFsdUmhlvIt0sKssbqxBUNjcMjsdo/KoYrLSvD6tdPOtsjYUyXE+F+mWOK1az9cgluNGuFtbOC8uNuYYbg4Qv2z7Cn5ARy61odzYy+bqli9s37qRvtKbfmB+UnPUjNJZaloLQ+TY39hIlvHkiOZG2IOMnB4A45ri/Dnh+xuJf7A1zTxbarZ3MeqzlSrpdkh0DDgfLyw244wKyddsLW+8Vyy29vHDYtqtpozLEoUSKu6aUHHYuFU/7tXyq9ibnpQ8QaCtmbxdTsvs4fyzKsq7Qx7Z9eKyJh4W1FNRv4dWxAqebfC1uiEIxyzqvXgfpXIePdLWy+IugPpentLAyi5vLW2jB3rC4AbbwCQJCKsavBPNYeNtUGjz6ZZz6GsUQmRULsolLHCk/3hQorcLncv4j8P6Vss5NRtbby412xFsbUI+Xj0xT1vtFuNVsLuO7hkurqF47RlfPmJw7Y/wC+Qa4Owt9U0C61W+1fThqsN7aW5jvMRxpGFj24OemMisfwNoxtvEHhN4X02dreOYXP2IEyR/usDzDnB57gDmjlVguezXNtFeWk1tcIHhmRo5FP8SkYI/I0+ONYo1RBhVGAPQU6isygqvb/APIcm/690/8AQmqxVe3/AOQ5N/17p/6E1R/y8h6/oyl8MvT9TR7UUdqK9U5gpKWkoAWgUUCgAoFFAoABQaBQaACg0UGgAooooAKKKSgBaM1XmuNnC8mqzSuRkk0AXy6jqQKaZ4x/FXn1/wDEezsPPFzY3KPDO0DK5UEsIy5xzzwB+Yqxq3jiDSdag02S0Z5J/uOJVCjjPzf3fQZ60AdsbqPPenpMsgyP1rOjbeit03DNOoA0gaWo4STGC3BxT80AKKBWVq2tDSm5tpplWF5nZAMKq9ck/Wr1pP8AabaKYKyiRAwVuoyKAJ6DRQaAA0UGigAooooAKKKKACiiigAooooAKKKKACgUUCgANFBooADQaDQaACg0UGgApKWkoAXtRR2ooAKKKKACiiigAoFFAoABQaBQaACg0UGgANFBooAKKKKACiiigAooooAKKKKACiiigArzv4syB18PWTD5ZtQDk+m0Y/8AZq9Ern/FPg+z8WfZPt088RtGZozE2OTj/Csq0HOnKC3aLpyUZJs5OjNaH/CqNP8A+grqX/f2j/hVGn/9BXUv+/tfC/6qVv8An4vuPc/tSH8pn5q54c0aG58Yw6q08iy20DIsa42uD1z+dSf8Ko0//oK6l/39q7pHw7s9H1SG+g1K/keI52PLlW9jXdl/D9bB4mNb2idvIwxGPp1qbhY64gMCGGQeoNZd3bf2cv2m1YrEGHmxdVwTyR6YrVpGUOpVgCCMEHvX1VaiqsfPo+x5dObg/IrAggEcg0tQf2U0fFtdyxJ2TAYD6Zo/s+5/6CEn/fC1xctZbw/Ff5mvufzfmT1Q1aDUp7dBo93FazK2SZYt6sMdKsf2fc/9BCT/AL4Wj+z7n/oISf8AfC07Vv5H+H+YWh/N+f8AkY+i+H7q01e71jV71bzUbiJYAY02RxRqSQqj6sSapjwSh8HppLXjC7S5+2reqvzC48wyb8fUkY9K6T+z7n/oISf98LR/Z9z/ANBCT/vhaf77+R/h/mK0P5vz/wAjlLjwhqd7Bq93qWox3GqXWmSWFq0SeXHArA8+uS2CT7ClRPFLaKum3mhadcw+QIZA94SJBjBz8veuq/s+5/6CEn/fC0f2fc/9BCT/AL4Wj99/z7f4f5haH835/wCRgeKdFvdX8OWMVrBCZrS4huXs2fEcoTrHn054+gqlYaZqGo+MLHVpdIj0aCxgljcLIGe4L4wDtAG0Yz9a6z+z7n/oISf98LR/Z9z/ANBCT/vhaP31vgf3r/MLQ/m/P/InoqD+z7n/AKCEn/fC0f2fc/8AQQk/74WptV/kf3r/ADH7n835/wCRPVa0PmaxcuvKpEkZP+1knH6inf2bO3D38pHfCgfrVu3to7WERwrgdT6k+pq6dKpKpGUlZLXp+gnKMYtJ3uSUUUV6JzhRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABUF0zKg2nGTzU9NZQ4wRQBnn3pGBKEKcHHB9DVlbX94c/d7U/7IvqaAPKPFHw9uhp2/S5Jb25nnaa58wj5nMUilx6Z3KMDjgVs6h4Wu5dWg8i6kNrdlze5iQ4wvyjkevFd59k/2qclqq9eaAMm3spIIYoxdSMI4wnIHOO9W4LSR23NO2B7CryworZC1JigCDyJP+fhvyFBt5CP+PhvyFT0UAZGsafPd6OLGOTf5rIkzueTHuG/8wCPxrVRQigL0AwKdRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUtJS0AFFFFABRRRQAUUU2WWOCJpJnWONRlnc4AHuaAHUVUh1bTrmYRW9/ayyN0SOZWJ/AGrEk0USF5ZERF6szAAUAPoqB722jg857iMRbxHv3DG4ttxn1zx9al82PzRFvXzCu4Jnkj1x6UAOoopqujqSjKwHUg5oAdRSKyuoZSGB6EGhnVBl2CjpknFAC0U2SRIY2kldURRlmY4AH1qNby2eFJUuImjkbajhwQx9Ae5oAmoqtJqVjFIUlvbdHU4KtKoI/DNPkvbWGNJJrmGNH+6zSABvoe9AE1FV01CzlR2ju4HVBlysoIUe/PFINSsSm8XtuV5+YSrjgZPfsOaALNFRTXUFuivPMkasyopZsZJOAPxNOeWONd0kiqvTLNgUAPoqOO4hlbEUsbnrhWBoaeFI2keVFRDhmLABfqaAJKD0opryJGu6RlRcgZY4HPFAC0UtVYtTsJ7jyIL62km5/dpMpbj2BoAs0VGbqD7MbjzozCFLGQMCuB1OadFLHNGskLq6MAyspyCKAHUUwzwhFdpUCuQFYsMMT0Ap3mIJBHvXeRkLnnHrigBaKgl1Cygj8ye8gjTcU3PKoG4dRknr7UlvqVjdFha3tvMVGWEcqtgepwaALFFIZECBy6hTjDZ4OelR3N3bWcYe7uIoEJwGlcKCfTJoAloqvb6jZXhYWl5bzlRlhFKrYHvg1ENa0pmCLqdmWJwALhck/nQBdoqvcalY2cgS7vbeByMhZZVUkeuCaBqFkbU3QvIDbg4MolXYPxzigCxRVaDVdPupPLtr62mfGdscyscfQGnRahZTSeXDdwSP/dWVSfyzQBPRTY5Y5VVopFdWXcpVgQw9R7U4MCSAQSOvtQAUUySeKJQZZUQEhQWYDJPQUgurcrGwniIl/wBWd4+f6etAElFMa4hVnVpowyAFgWGVz0z6Un2q32hvPi2lgoO8Yyeg+tAElFVJdX023laKfUbWKRThkedQR9QTVhbiCS3WdJo2hYBlkDAqR656UAPoqP7Vb7Q3nx7SwUHeMEnt9aZLqFnbyGOe7gicdVeUAj8DQBPRUT3lslr9paeMQcfvNw28nA5+pp/mx+aIzIvmMu4Jnkj1xQA6iori7trRQbu4igDHAMjhcn8aJ7u2tYRNc3EUMR6PI4VT+JoAloqNbq3YIVnjIkUumHHzKOpHqORz70yPUbKZisV5buwBJCyqTj86AJ6Kaksciq0bq4YblKnOR60k08MEMks0ipHECXZjwo96AH0UiyIxAV1JIyBnnFKGUsVDAkdRnpQAUUEgAknAHUmk8xPl+dfn+7z976UALRSNLGjojuqs5wik4LH29ar/ANqaf9p+z/brbz923yvOXdn0xnOaALNFNSWORS0ciuqkglWBwRwR+FNF1blWKzxkKMsQ44HqaAJKKglvrSEKZrqGMOMqXkA3D1FIuo2bhCl1Cwd/LUq4ILYzjjvgUAWKKSSSOGMyTOsaL1ZjgD8ahTULKRGaO8gZVxuKyqQueBnmgCeioJdQsoJCk93BG/Uq8qg/kTU0csc0YeF1kRujKcg/jQAtFIrowJVlYDqQelHmRiPzC67P72ePzoAWio2uYElETTxrIcYQuMnOccfgfyNOeaKJlWSREZzhQzAbj7UAOopvnRb3TzE3IMuu4ZUep9KTz4QIz5qYlOI/mHz8Z49eKAH0Um9DIYw67wNxXPIHrik86LzGTzE3oAWXcMqD0JoAdRSSSRxRl5XVEGMsxwKd2oASigkKMk4A6k0hdAm8soX+9nigBaKC6rjcwGTgZPWoJb+zgl8ua6gjk/uPIAfyoAnophnhy4MqZjGXG4fKOuT6VDbanY3shSzvbedwMlYpVYgevBoAs0UFgDgkfnUSXdtIrNHcROqrvYq4IA9fpQBLRTTNEGQGRAZPuDcPm+nrSPcQxSLHJLGjv91WYAt9BQA+lpkk0UWDLIiZ6bmAzRHNFNnypEfHXawOKAH0UUUAFFFFAHCfEO4u7No59PjmluXjEMdrFLj7cGJ3IAOVKgbt/bOO9VEWbV/h3qSyPc3UuojdJdx4IR8AsNhI2Km3bjOeM13F7oun6hOZry1SWQxGHc3UITkgelQXfh2xudK1Kxij+zLqSsJ2i4JYrt3fXgUvstD6pnn3hTUZzr4JtZ5BC5CLGSxlTb94Av0yffpUXi+2gupLzUtGtrqKxgiEDTW4+SeRpULFsn7o2gZHUk13Nr4buP7Usbu/uoXFhu8lYIBHuJXblj9D0HFamqaTBqujy6dLmOGUAHyxjGCDx+VV1T7E9zzS60c634WlgMUECRapdTPqFyflj2XjssYA6lmAH0PrW34U8y68YQ3MZCwQ6bJE1uq4FuxlUgDIDEHBIJFbsnhKE+H5dNjuZF3XrXqSlQdshmMoGOhAJx9KsadotxDrUmq6ldLPctALdViTYqoDn8Tk0o6P+u1v6+8ctU/663MuwvJta1S7j12cWKW+9RphO3fHyPMdv4gRzxwOO9c3OiXl59p0qNrTQb+8tbABSQLkKzs8g9FbhAe4r0HWNDsddspLbUItyyRtHvQ7XUMMHDDkcGqf/CJ2Lae1lJPeSwYXYslwT5RUgqy/3SCBzQtGn2sN6po5rU9PgRfGGlxb0sobCO7jiRyAkhjkBx7fIpx61PLaPIdHvNRsZ9R0xdKCGOI58uXgl2BI6rxntj3rdPhKxfT7y1knvGN8c3M3nkSSjbt2lvTHGKkbwxZyaLFpUs95JaR4BRrhiXXGNjHuvtS7/L9V+v4A9bfP9P8AI5jTY7zVvhvo94VMkVvc/aWhuJNvm2ys4QMx9FKNz120y+sLq3Vb17b7NBd63aSw2qNu2AAAtxwCSM4rtdQ0qC/0G50kfube4t2t/wB2MbFZdvH0BqnbaRqcHkJLrJmhiK/I1tHyB744+tV9q/mn+K/yF0+TX9fezJ1XWLBJo4p/DV613es8cG63T53CFuu70BrJ8yaDwbpei3dtJaSW9osVxNc2xkaPCgFkxkZ68n0rt7/So7/UNNu3kZW0+ZpUUDhiyMmD+DZq5PEJ7eSInAkUqSO2RioezsNbo8vt4DceCrrRdDP2yKa1T7Fci3ZJJFBBAYnhvlHB71m+Ji17pKwWd0sTRw3W5nEEZbdAVCbVOOTxnrzXrOkacmkaLZ6dE7SJawJCrt1YKAMn8qq6z4dsdX0W8sDBDCbqFovNWJdy5GMjirluxR0sec+OIV1G4t0vbOaERyQsmxFEjhSMuSW5A6gYAPQ10+nm6u/C8drBosU7F/3TSDZDgg4kOGOfcD1rVu/Ds/8Aazajp9zEsslulvIlxEHXCkkEeh+Y/WnWXhCwttLt7O4L3Bh3HeW25LMWPA4AyTx2pdGhdjA8Fxx+HdG86dYrmwSN3XVIwQzDdyjJ1zngY64rJur68TwT4ktNRtJLO4u5ZL60EgJDRPICAT2YZwRXc2XhLRtOMf2O0EYiOUUMcA9elW9b0mPW9HuNPmdo1nUKXUDK8g8flTGYF3qWpaRqdosNwuoXN0UE+nRgnYOAZFJ+4AOSDwecc1oeMEE3h6SJ0mEZZZGliKgxbGDhju4xla14rOCCSWWGJFmlxvkA5bAwMmsr/hH5Lrwe2japfS3Ekke17knLZzkHnrjjr6UnsC3OZ8M+KdXvtTeK7kubgCRjHD9nSPfF0DHIB4zziqtlczRa7Zy688YvrGZy6R+TEilgVySBuPytnBrr7bQrt9YtNQ1S8SZrKN0gSKIIPmwCW/AdOlQt4Vkj1O/u7TUPKF9MJnR7aOTa2xV4JGcYUUdg6M4e7Rn+H+m2eoW6rZLau01wHUuV6sIzuxnHfn2rR8I3s2kaLKNI08Xdt5KeRKzqrKMcea27AAHfGetdnpPhyz03QbTTJlW8S1TaryoMn146CoJPCdo02rGJvIi1OzW0eOJQAgAcbh7/AD/pQ9Ex72ucpPY6vYeEtI0e7Yeda6lbCK+i+ZJVLEg4PcdxV67utQtvGMerXM9pEtvEbP7HLKFeWNmBM2egOQML6V0FpoupW0dvE+stNDBtARraPkD3x+ta0lrbytulgjdvVkBNPZ6d/wBEid1r2/W55rc3c0Phl4YfLCz6tcXC3DBXQr9pZwoO4ckDBHoaboWvjULXVJWCLb6nATG0Ftg22YwgyAxYLkZ6dTXUP4Ji+zW8cN2Ymtr+e9jPkq65lLkqVIxgb/0q3pfhk2Gutqk14JpDb/ZwiW6RKBu3Z+UcmpS0s/60KvZ3Xf8AU8+W6aeytdJN/debavbSS7rmZ1CpIrZ2bOhCHFdd41mSW0067CzCO2lFxHMsasrEoyBSjdThs4xmuig0mODXrzVBIxkuoYomQ9FCFsEfXefyqPXNH/teG28uYwT2lwtxC+0MNwBHIPUYY022/vEkl9xyHhPWtSvLKQy2ImuTEFniSJIBGxHrjdjrjIrmjcNHaJosySR2dukJE7xIoOH4XzPKznKjJ969NsdCmj1C91C/uxLd3cC25MSbFRF3EY9TljyawY/h7ILeOC4v0ukQLuM0eS+O5568U1uHQoa9Jqdn4hOom0s4pbm2SLyZpRMcIWO4AISPv9fapbzUb+Xwbtn+zWvmyo0c9qQ6OocFlO1RtOMjPrXUX2iXLa0uq6ZdJDcG3Fs6yx7lZQxII7g5Y/WpNL8P29ho7WFwftSySvNKXUAM7uWPA6DJ6Uun9dw6/wBdjgfDN9ay69Dd3FpcCWBpY4l81nLA8ZxjGDgdemalEZ0zxBe3N+8Nj/aAV2htwTJYBVwuSAR8wHPTmu1bwxZJrNjqFmi2zWhfKxjAk3Ljn6VDdeHrv+0dQudMv1t11IL9oWSIPghAmV/ADg8UpaqwI5FU1Pw9pOi6o8l3FaWegxxXD2sSTFWXBOQxx07itvRbbVP7O8TXN6LpmvCXtzOio7AW6r91eByDWmPBOitpsdlcW7TxJCsJDufmVVA5/AVoafodjpkpks4ijFdpy5PH405K6a9fxdx395SPNZzax+HNDj0/ZG5ltPtaPDIJGcMM/MeOppNDS3XwzJ/bVvPM8+k2404pGzZHlD5VI+63mZPbqK9L1fR4NYtYoJ2ZFjnScFP7ynIp+jaYmjaLZ6bE7SJawrErsOWAGM0bqV+r/QS0at0X6nmesR3oe7S8ZTcqNJ+0GTJTO87i2O3rWlq8gS002Kyn01l/tS2aaOzRvm/eDHJJrtotGhj1q+1FmMhvI4o3jYAqAmcf+hU3U9DttRt4IgBAIbmO4BjUDJRtwH0p396/nf8AL/IOlvK34M8w1lbv+3r+Se23GWYvtl3kxj+6Cq4xXc6DPYJ4BtjqMEMGnxQqqhWLIyADaR3/AA61Dd+CJrjUru5Go7hcSmQLKmdmf4R7Cuh0rS4tN0a208JG0dugVVVMKMdMCkvht6A/iucDd2Fxb6SNQtrYWh1DXLaW2tpcgIgARdw7E7dxHvWjr2rXVvqtoLvR9PL5LXIabezptIBGEJXnHX0NdZq+kx6vFbJLI0Yt7mO4G0dShyB9KpXWh3a61c6jpd4kL3caRzJLEHHyggEfgenSjpbz/RL9A6/L9WcPcaO0mkaVZ6lGYNLYSs13buLjcxYOgwBwMlu38IrQ0CLUYvE+gq8aQyf2bcGVZgdxTzkwcA8MRg46Cuj/AOETWPQdO060v5oG0+YTRzBQSzAMDkHjB3Hir9lp97Beefe6j9rwhUA26IRk+oGe1NaP7/y/4IPVf13MHU8W3jiV7qyOprcWK+TAuC0IViGO08YJYc9eKy7G0lt9F0TV5Ps15ZxeewtpJsIqysGjwWHO1Rt5Heuqn8I6NcXzXk1puuGBBk3nOCckfTNWNN8P6ZpNr9nsbVEhwAIz8ygDpgGpW39eYPc4G006SXWtINvJaXJtxfXMltHLu8tXkjZY1wRnjj0qN4W0y/1X7dLDaPqIMrJag77E+XtVGYAjDAKT05yelegSaFbt4gs9UiCxNawyxbEQANvKHJ+mz9aoXfhm6e41P7BfrBBqpzcq8QYqdgQlT7gDrSavG3k/zGnZ3OYmTU/D9npOqO93DaWmirFcNbRJMVZSCchjjp3FJJFqRim/tM3Hkatrdqoa4RUZohGpI2rwAWjx7110vg3Rri0FtdWxmj8oRMHc4ZQMc/gKjvfB2ny2DxWam3nBR4pSxbY6MGU4+o/KtG/ev5/rclLS3l+ljH12G2l8RQ2vh+3Av7WX7Ze3KE/ulAJ2H/afpj059Kpafapb6V4R1GF3+2alKI7uUOd0wkhd2z/wJQfbFdTZeFLOwupri3ub2N7ic3EyrcNsdycnI9Pb0pLXwhp1pPG8UlzshD/Z4jMdtvu4JQdjg4HpUrRf1/Wo3r/X9bHK2emTNpF3bWkU11aWuvym4tlcl5YRuG0ZPOGKnGe1V7OaY+JJbnTrC6TS9M1QbrckBkZrcqwAJxjew4z1JrtbDwta6bDeR2t3fj7Y7PIzXLEhmOSwPY+9Fz4Ws38Nvo9kzWkTOsgkT5m3q4fcc9SSOc0LRfJfp/kD1f3/AK/5nMz2k8vj/RdSks2tllupQFkTDkeSeTgmob/Tp7C++yTpDs/tQ6h9pSB2kKGUybMj64rsrXTtRiuo5LrVjcxp/AbZFzx6gZFZmr+DjqWuTaiLlD50aJ5U0e4Jt7j60bNNdA3Tv/X9XGeCYYRpepXZuWliuL+6cqznZGPNc4wfun1rJns7e8fWtW0q3+z6fb6VPbxuAVFw7AlmA9BtAB+tdhoukrpOlizAiK7mYiOPap3HJ4/GrF9ZJe6XcWJPlxzwtCSo+6GBHH51Mlpp2/Sw4vW773/G555rkiXnhe2tdQt7mPTodPR7meCBWlKBcnaxbge+M1NdG5g8CaistpNELSNbq1uXtxEWZCGXdhiNxxjgdK6K58Iea8Bt794hHYixkRoldZY855Bqe60C7vtMWwvtR8+Bpo3kHkqmUU52DHqQPwzWkndtrq/1/pkRVkl2/wAi7qFwU0NpTZtesUH7hQPnJ9c9B6mvOptG1TTfBE9vK0Sp50U/n2iqyXCvcK2wnqNuccdRXomnadcW0F5HeXb3AuJpHQMc+UjdEHsBWdp3hvUNN022sINbdra3RY0R7aM5UdMnFSt7ldLGPr2tHTriH7dYaTLLNMkH7y4AdQRwzA9BgVp+ErFrfwmbPUXtJrYb/wB5BLuRlJJbnsBnFdBJY2szb5raGR+7PGCTWXa+F7SHw3daJK7vbXTzF9h2ECRyxAI6Yzijow6o4m5RLu7NzpCNaaFe3dtYDaSBcgMzO49FPC57ipPEHmWN7rvhzQoWklvIbeW1tlJ2o5J3nP8ACMID9T712I8JWP8AZ7WUk95Lb4UIklwT5ZUgqV9CCBzVvS9DtdKeaWNpJridg0s8773bAAAye2B0o6WDrc85+1vfePLG5aWRxJNaEJJHsaL5bkFCPYjrW3ryW66zr39tW88zTWyLpxSNm42fdUjo2/J7dRXQXvha1vdfh1YyyJNG8b7VAwdgkA/9Gn8q3KN1b1DaV/JHnVsmrnUNVtAI/tk2mWkVxPK+Fgyj7nPrj0qC1ee+Hh3R4L5Xk0+7yl3EpBliWF13YYAZ5AOM+tdrP4et7m81WeWSQjU7ZbaVBxtUBhkH1+Y1UtvDd19q05tQv1nh0wlrdUiCFjtK5Y/QngU73bf9f1/mHRJf1oVNanm0PxUNSWe3Iu7NLdYpd27MbMxYBQc/fFc3qn2jVG8R6jBdF5Bp8CmG1DjJDMfukZ6f1rutX0R9R1Cyvra9e0uLNZFUiNXDB9ucg/7opNK0KWw1a81G6vmu57qOOM5jVAoTOOB/vGkth9TjfGl7/aelhtPvZPKj8jzIj5qA4lTPGAp610Hj61Sfwa8zNIHt3idNrlefMQc461t6zpUes6TLYSu0SSMjFkHI2uG/pUetaDba9YCzvZblIeMrBMY92CCM468gUC6mX4nH2m48P2EpP2a7vMTqDjeFiZgp9sgH8K5qxt45/iZdeHZsvpNqZbiK3LHbvMcJx9AXY4/2q6qfwr9rvD9qv7p7WOOP7OpmJeKRS37wN6kNj6U//hDtPWNGjlukullaX7Wsx81mYANlu4IA49h6ULe/r/X6g9rf1/XQ4/wtaR63rms6bqjPPaaWjwWis5+VTNKNw9wFUZ/2azVtb+58F3V3fwW07axbQpBdTTYdWESr90KSfuk8V6A3gvTBDGls9zbFIjC8kMxVpUJJIY9+STn3NT3/AIdSW304abILSTTG3W3y7lA27dpHpg0ugdbnCafYyJp/iMQRRTWl0kUTSWzNFsITB+VgWznHGOal0bUrpvE2mvdW9qJkV4LeOEmHzMgEliYxkgLnH1rqpvCUt1p+ppc37C81KWOSWWNNqrs24UD0IXB9cmmad4OFlqlveSNauYGLKVgwwJBHBzx1qlvdie1ihv0228b61PrUbhw0LWzmKRgAIlztwMdc1ynhJdSvNFvrWJrdI7vTLdHuixUW8JRssQerYz6CvYiobqK5qz8D2Vnp97ZLPM0N5ZpaNnGVVVIBHvzU20a/rqV1uc7aTXGq6h4dsUvF87T5ZM3MSlfOjEeA2GA69CBmr95pU6ya1Fe6RNqNzeSM1rcoRtVSMICc5Tb7ema2IPDlzJf2c+q3q3CWKMkKRxBN24bSW/DsOKs/8IpoZ/5h0f5n/Gm9RLQwfEUGpWulaSlzE11Db27m9njhWVlZVXBwxHB+b8qn8E2KJc3WpRC7CXkMW0TW6xIVGSCoUnk55/CtWTwhossbRtZgRsCpQO2CD171Np/hzTNKkR7GAxGNdqgOSAPTFO+rbFbRJGpRRRSGFFFFAHH6/wCNZtI16Ozjs1eCJs3Uhc/KpX5TwMLz69q1tG18anq+q2EoiilsZkRYw+WZWijfJH1cjPtWF4jtbJPElxaXk11a2WrWm68aOMukxUhNmQCVJU9uwrV8N2tmdS1m/tJEmW8uEZT5TKyKIY02kkDPKZ/GiO1/63B7jB4yg/4Si/0pYmnFtFGYzCMtLISd6DJAJUbSQOeaoxeNbq40jRrpbe2tn1KSZGNy5VItmepz1OKydVuBba5c6fZQG1g0+ZPJBZUXcUVty5Un+LGc9QaWOBLbRdEh1WScJcJMJJLNm8qNgSVO1R1OTzjtS+zcf2rHTaN4hutQ1i+09xZzNbW0c6S2sm5GLlxtJ5wRs/Wqdh4tv73xVc6GLWzE1qqPIwuM5BJyF45K4GR2yKp+HbGBNDQRXN5f6iLZUuollaNHYjDEZUDuSKbFpN4L6SN/DSLp8SRtaJHcKHikBbe27Gefl/Kq2ZPQ6l9YCeKodG2DMlm9zvzyNrquMf8AAqxLbxk9/HaW9qsBvrjUprUxq2fLjid9zEdiVT82FSp4SkuNQXW7i8kj1neCkowRFF/zxx0K+vqea5/TbWTTJdJ1VrVgI9U1GGUbNrHzZX8sn6lVAP8AtClslf8ArX+kV6f1p/TOgv8AxhLok2zXNPEAe2lni8mYSE7NoKkYHJLDBqxB4h1B7x9Pn0pY7824uIYvtA2yJu2t82OCpIz9RXL/ANmanruk6ldalpt4muXkIMQlQCK3VGDrCpyepXk9zVzVLzXLjUv7Z0fRroS29j9kjSZAG3yyIWbbnkIEz15zR1S/rr/wF/Vxf1+X/BNJfGU76o2jppwbVlmMZiEw8sKI1cvux6Oox6mrGkeLRqusQ6f9lMMpinMys2TFJFIqFfcfNnPpisOx06TR7/S9St9N1GbyBPHfPNGPOmebYxlxnnmMD2BHpULaf/Z8cGsajb3dtNf6pNvaF2V7eCTLDdtz/wA80yPU0LdX/p7L/MOn9bf1ob+n+LJb7xc+jm1iSNLV5/MWbeSVcLjGBjrVHSPGN9fPoj3AsDHqyMwihZvMhxGX5yfbFWdGtNKfxNFd6XdPIYrOSFkkjfc251bcWYDPSucurkW+v34sIPsZtp2iQlkTPA5X5Dwc+tLXlt1s/wA3/wAAfVvp/wAA7Hw7r1z4g8JW2qR2ZhuJoVcRSfKpJAPB9OetZ114vvrKPVlubGFZ9OFuflkLK3mvtx7EDn8RWj4ftryDwNYW0ElvHdpaIgdEzGrAAEgccVl+INCGm+Bb+KAy3d3cTxSzzMMvM/mpk4HoB07AVcviZK1SJPFHiy80S5SOwht7xmZd0KBmkjU/edgvQAc+p7VaPiKaOPSbmSS1ntry4NtM9uSQjMDsP5jBB7sK53WR5es38EV9f2E6TWzrMJJCJoiVMgGARwNwAq5e2dnLa6XZ6PPc3Iu9TilPnMx2LE3mO2CBjoB9WFTH9f8AIb/T9DZ8U6xqGiWpubM2UmQFitpQxlnkPRVwe/H0qtp/i261HVtUsrayhY2EMUg/0gDcWLg5PQY2VW8ZPHp+q6fc29uwu7jev2lSAECgcEkHGc8fjWXpdlqmqJrq2klmJL21S3Dvyy8SAklQv94Y/GlrZj00Ne48bT2t8bW8sIrZ47q2ikdpw6bJd3zA8cjb+tVNV8b6hba5eWdlFaGO2cJumBBbKhsj5hkc9aqap4d1K3XS/tcVnOsuoWq3PlI7MwRSoJycAVk63Y2X/CQ30EULxrbyCP8AeIvPyg5BKkkc0/8AN/oL/JfqeiaTqGo6h4YgvZLe3F9ImTEkn7vOfWsmbxfe20eorcWMSzWN5a2zbZCyt5zoD+ID5qTwrZXcXgiJbVoLa9mBJcDcinOMgfQfTNQa9oa6Z4PFvaiW5mk1G0mnlYbnlf7TGWc/gPwApv4vmvzEtiXxZ4tu/D1xYpHYK0dzeRwCR5gu4MDnA7fWugS4u5dNMy2oiuSDthkfIznuRXJaxpuoXd/9og0H7Q8kyrO1zcBwYOdyoCPlzweK2bGzv18PzWNmkmmyD5YZJ5fPKKTzj6DoPpSXw/15DfxIzF8ZX/8AwjN/rE1jbxR2lw1vjzcgss/lMT6Dqavap4qFneJDaPaTBolk5dicHOD8oPHFYms+EBoPhS/t9Jllls5zC0lq43lpfNTdID/tDJI9eak1XR9Tl8Uw3unafcQW4t2gnNteCLzgpBiOAOMZf86GHc1LTxa9x4budSngjtmhuTbr5pYLJ8wAI4zznjimab4uuL3XrXT3tEZLhXJli34j2gHnIHXNULLT9Xm8KW+ma7pl1dzqUllm+3Dd5ikMCpxxgipPDlpr1jcF9Str6ZjcSbC9/uRYi525XHJC4/KmtxPYg8T+O77Rdeltbe0ikt7MCS5ZnAJRl4wCck59Kv8A/CbbLTX3ltg02kEYjQNh18tGyTjHBc/gK53x5aY8WW5iEzyrF56ySysEUksuFwMdunbNPsbWefwPrtzdwySSvMzNHa3D/v18tEO/ueByB6e9Svhb8n+aKduZI6Ow8U3EmtW1jPJYXwubeSYHT3LGPbj7wJPXdgdOazn+I3y6mY7FwLeaMQGQABo2ZFZmIPZmbpxwKytE0K2j8SWcVveSXXnWrq01pI6G3UbSFPbBPY88VLrPgu/i8PW8UzQXEenwpbQpbxEPMpmiJL/QJ296rqSdfo+uzar4dk1FLeNpFmmjWMSbVYJIyZ3HpwuayfDHjO813Wb6zksbdI4LjYjrdAkx7R8yjHzjdkZHFN8OaG934I/s8s1mn225JRoQ2V89yBhu3Q0zTNG1DUNQvFu71Y10u7EVowsY1yuxWyDj1Yjjjikvi/ryG9ix/wAJZexeLbrR54rVhawpO/k7mkZWYjAH94YBPswrY8M6y+v6DHqEkQhMkki7PQK7KPx4rmNRsbay8QCyM66ZDbxrdxXjwiR55mLbvmPpgcd81reDLsJo9lZPZTwSywvcsxQ+X80hzyehJOcehoWwPcpz+O5ovDtxqzWEMEI8wQebcgmVlfbjaB+NSad4wvtXmNpp9jC92jTNJvkKoqJK0a/i2w/TFU9c8J3dj4Lv7S1vXu4VV3itzboWyz7sBsZ6n9KXwx4enks3v7e6m029Fzdws3lBt8ZuHZchvTOQfekN7E1z8QPKsvtcdmPLXTZbx0dsMrxuEKZ6dSefap73xjLYeH7zUZzp7GKMGJIrkPvb+6cVnX3hJH1SLQ7cyLbyaJPAbhl3fM0qHJPqTk1N4m0bUU8Haikgs5FEB2xW1rhmPGPx+lPpf1/Nh1t/WyOp1LWLbSrJZ7reWkH7uONCzO2OgAqh4Z8SrrdhH9shNpqAj3zW5BwvurdCvuDTtZ0e71GGB7C8kt5UUKQZXVcY9F71z2keHZLPWI9A1BpZbFbLegjnk8vG7bsIY/jQt7C6XLMnxBQ+G9W1K0tGuZLGWeNViGVwhO125GFYYORWnrXiK+0e0nvDpkc9tCiyMwuQrFTgZ249TXJauFlttX3eVbXEReyh0xLbDXEIOFG7GfmHOR0zW9qPhzWNS1iW4c2JtJbVLcQvvzhXLgnB69BQun9dBvr/AF1NjQdd/tibUYmEaPZ3bQBFbJKgDBI/Gsm68W6hb+KodCNrZ+fNE0ik3GAMEYU8cMQSQPY0vhC1ubSPXp7m1RrxtQlP7tSgkGBjBPaqD6PeG8SFfDSnTXiczo1wDI8xZSr7sZyBu/Oj+X0X5C7+r/M6fUNYNhqmk2TxqW1CR4yd33dqFuPXpVNPE4aTxDEwh83SCdkYf5pFEKvkj6tiqknhJ9ZkS+1Kaa2urdQNPVWybMj+LP8AETjnPbipPDdh/wATfxL9uhEjS3aqZGjx5q/Z4wce2c0tXdeT/QasrFX/AITqVo4CumTRGXTJ77M67VJjVCFGDyDuqOL4gvJHcsNJun8nTkvPkj6Fg+QcngfJ1rF1C0tbJ9cNusOnzW0cltbWa2+5riMoOAx5+c8fL04rT8T29vZR6FI+jRXCSL9nuBIGOxAhKqWAPfNN6q6/q9/+AJaNJ/1a3/BNfw74tbX9VNqLRoEWwhuiXUglnZwQPb5etE3iW9ttYt7SW1t3Sa7+ysYpSWjPltICR9APzql4a1JtT8W3oFglrDFp8MaSRhv78ny7iB04496pWUWkW2sar9vudTtp0verSysJsIoD8Ajpx+FEt9PP8xK9n8vyLKePWuPD9pf272X2ia3M0lsd7MCOw2/lzV7RfF8mreIINNa0EJewF1JlssrEj5cduvesRdItNC0XQLlJrjcFFlLdWqMoCEM28rjPVQPxrX0a006TxUl9YX0lw6WjxOJo33sS4OSxA6dMU1v9/wCT/wCAN7af1qv+CT634wj0efWYnRd2nWK3SE5O9mD4UgdB8nX3qpH45kFrHNLYmTzNS+xBIeWx5ZfdgnrxUGueDpF03VtUju5ZtUuLSdZyq8ToYyEjC9gvGPx9ar6B4duNT0+/guJmtoxfM/lyW6vuO1cMNw+o4qY7v0/X/IJdLef5f5nQ2nij7Z4lh0pbC4gEls8++dQv3WUYGCf71OPiMtpevXUcADaS0qAMeJCibqxfBGg3S2dtq09yIbhwyyRi0RCV3HjPUZwDU+t+Gr2DR/EL6bqcqrfxzTNbiBGy5jxtBIzziiWi+T+++n4FR1l8193/AA42Hx002gajqSQwOLS4hgVlfKDesZLMewUuc+wq/pHiKe78QNpc8ljdH7N9oE1i5ZU+bG1sk8nqPoa40aQY/DOtrNbXCobqBGiQsiFWihVnZV+8ByePQ1d8O6JAvieSGC+nuBJaBnvLSZ02bW+VG7H7xI79c1fX+u39P8DNfD/Xc1rjxjeRwXt8iWQtrS6a3Nq7n7Q+19uQM4yeoGORitzxFq1xpHh651G0tkmaGFpCJJdgXCk+nPPauQ0+PQb2NL3U7nUrW/3HejNKWUg467fbrW74nt0tvCSTxLLqBtpI5IxcFpN2XA3MBy20EnHtUP4S18RVs/G876boF3d2caJqsmwmKQyY/dlsgAccjGD0rZvdT1e1E8kWkxSwxBmDm7CllHOcY4rkrSRB4n0oWsVrc+ZcyTMLe3liEb+Uw3ZIwM9PxrXvLbX7zxALuTSE+yNYyWrxi/AJLMDn7vpTld7f1p/mKNlv/Wv+RJd+LL5NN0a+tNPiaHVWiRFkmwyM6lsH16VaXxL5Ws6hp999mgezso7kbpMbiQxb8Bt61Ql8ISa3Fbx6rusbaxRRY20Eu4wyDpKW4yw7D6+tV1024fxBrUd/Z/2k8unW9u7geX5gIkD4Pbg9vWiXW39dgj0uS/8ACcXAmtlW0hnWeeOLEJckBmA3crjAzmma54z1Cw1+6sLKG1KW+0Fp1IJJUNwdwyOazNL0jxTZvpnnRXzrbu5uE/tDKuv8CgY7d6oeIbW1k8RXcX2eSJo9m4SKpBLKG4JByOfzoBHeaRrGoXvhddRuLW3N1lsxRzbUwGIzuOccDNc9pnxBu7nwfDrVzYwqrXcdu3ly7/vSbT8oGQQOfenaTp234aXjy2yyy7JZfIiY7ZdhYopAx1wMgVlyzpI1jKkVpLLcXdr5sEFpNGRhxg5xj5f6U/t281+Yvs39TpPEni640FrSQW9u8F5KkcRklKsNw5YjHAHc10llJPLaI92kaStyRG25fwNclq9jqM18bix0BXmlnUXEs06tvhGQVAI4zxW/oEDWli1uNPawiRjsjMvmcHnj0HtSWw3uatFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAEA9RmgADoMUUUAMaGN2y8asfUqDTgiqu1VAA7AUUUAKAB0FFFFABSMiuuGUMM5wRRRQAtFFFABR160UUAAAHQUxoYnbLxox9SoNFFADgAqgKAAOgFLRRQAEA9RSbF3Bto3AYBx0oooAGRXGHUMPQjNIkaR52Iq564GKKKAHUxoYnbLxox9SoNFFADlUKuFAA9AKWiigAooooAKKKKACiiigBrxpJjeitjpkZoVEQYRVUegGKKKAEjhjiz5UaJnrtUDNPoooAKKKKAGyRRygCVFfByNwzinUUUAFFFFABRRRQAUYGc459aKKAGGKNpBI0alx0YryPxp9FFABRRRQAUUUUAMaKN3DtGrMvRivIp5GetFFAAAB0GKQgHqBRRQAuBjGOKAAOgoooAKKKKACiiigAIBBBGQetMjijiBEUaoDyQq4oooAdtHoPypaKKAEwB0ApaKKACiiigApjRRucvGrH1Kg0UUAOVVRcIoUegGKNo9B+VFFAC0UUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQAUUUUAFFFFABRRRQB//Z)

그림 전국 반려 가구 수 그래프

### 유기동물 현황

* + 동물 자유 연대 「2016-2020 유실유기동물 분석 보고서에 따르면, 유실·유기동물의 수는 증가하는 추세를 보이고 다양한 동물이 유실, 유기되는 실태이며 특히 개의 경우 하루 평균 200마리 이상이 유실, 유기되고 있음.

* + 동물에 따라 계절 변동성이 존재함.

※ 2011년 96,268마리에서 2014년 81,147마리로 감소했으나 이후 급증해 2019년 135,791마리를 기록

※ 개의 유실·유기는 지난 5년간 총 421,610건, 일평균 230.8건 발생함.

※ 개의 7, 8월 발생건의 비중은 9.7%와 9.8%, 일별 발생 건은 262.6건과 267.3건으로 평균 대비 13.8%(31.8건), 15.8%(36.6건) 높으나 전체 발생 건 중 7, 8 월 발생 건에 비해서는 상대적으로 비중이 낮음.

※ 고양이의 경우 발생율이 가장 낮은 2월(3,811건/2.7%)에 비해 가장 높은 6월(19,729건 / 13.9%)이 5.2배

* + 전국 유기동물 발생 건수는 다음과 같음.

표 전국 유기동물 발생 건수

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 지역/발생 건 | 2016 | 2017 | 2018 | 2019 | 2020 | 합계 |
| 서울 | 8627 | 8588 | 8008 | 7271 | 6183 | 38677 |
| 부산 | 6962 | 7414 | 7493 | 8097 | 7470 | 37436 |
| 대구 | 4003 | 4186 | 4805 | 5469 | 5042 | 23505 |
| 인천 | 5621 | 6074 | 6874 | 6799 | 6646 | 32014 |
| 광주 | 2607 | 3672 | 3248 | 3830 | 3556 | 16913 |
| 대전 | 4011 | 4498 | 5333 | 4846 | 3217 | 21905 |
| 울산 | 3084 | 2885 | 2903 | 3083 | 3334 | 15289 |
| 세종 | 384 | 345 | 404 | 494 | 330 | 1957 |
| 경기 | 21679 | 22917 | 25681 | 27930 | 26931 | 125138 |
| 강원 | 3929 | 4747 | 5312 | 6179 | 5488 | 25655 |
| 충북 | 3836 | 3546 | 3731 | 4905 | 4784 | 20802 |
| 충남 | 4572 | 5646 | 7272 | 8493 | 8852 | 34835 |
| 전북 | 3588 | 4510 | 6024 | 7838 | 8819 | 30779 |
| 전남 | 2752 | 4214 | 5863 | 7704 | 8441 | 28974 |
| 경북 | 3693 | 4861 | 7477 | 9128 | 9638 | 34797 |
| 경남 | 6583 | 7919 | 11383 | 14140 | 13683 | 53708 |
| 제주 | 2626 | 4818 | 6886 | 7307 | 6303 | 27940 |
| 합계 | 88557 | 100840 | 118697 | 133513 | 128717 | 570324 |

### 유기동물 보호소 현황 및 실태

* + 농림축산검역본부에 따르면 동물보호센터의 역할은 유실, 유기동물 구조 및 진단, 치료, 안락사이며 보호, 관리 및 입양, 분양을 수행하고 있음. 또한 동물보호 교육 및 홍보도 수행하고 있음.

* + 보호소의 기능 구분

표 보호소 기능 구분

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | | 기능 | 비고 |
| 1차 보호소 | 지정동물병원 | 유기동물 보호 (3일 이내) 진단, 치료 | 각 시도의 유기동물 발생 상황에 따라 1,2차 보호소의 기능 통합가능 |
| 2차 보호소 | 유기동물 전문보호시설 | 상태가 양호한 유기동물의 보호, 입양 |

※ 참고: 농림축산식품부 동물질병관리부 동물보호과

1) 지방자치단체 운영 동물보호센터

* + 지방자치단체가 운영하는 동물보호센터 중 직영 동물보호센터가 차지하는 비중이 지나치게 낮음. 민간 위탁보호형의 경우 시,도가 시설의 설치까지 관할 시,군,구에 위임하면서 경비의 일부를 지원하고 있는데, 시,군,구는 시설건립 예산 및 인력부족, 민원 빈발로 대부분 민간에 위탁하고 있음.
  + 2008년과 2017년을 비교했을 때 보호소의 수는 증가하였으나, 민간 위탁보호형의 비중이 전체 약 90%로 여전히 낮은 수준.

표 보호소 현황

|  |
| --- |
| 2008 년 25개소(6%) →2017년 40개소(13.7%) (민간 위탁보호형의 비중 전체 약 90%) |
| 2014년 약 2만4천마리 (총368개소), 2015년 약2만1천마리(총 307개소) 평균 보호기간,  2014년 20.6 일, 2015년 23.6일 |
| 유기동물 법정 보호기간은 소유권 이전기간인 최소 10일 |

* + 유기동물 수에 비해 이들을 보호, 조치할 수 있는 동물보호센터의 수용 규모가 부족하여 위생 문제 등 관리의 어려움이 빈번하게 발생함.
  + 전체 동물보호센터의 유기동물 일시 수용 가능 마리 수와 평균 보호기간을 고려하면, 보호센터는 그 수용능력을 초과함. 전문가들은 현재와 같은 수용 규모로는 유기동물에 대한 적절한 보호 조치에 한계가 있을 것이라고 지적함.

2) 사설 동물 보호시설

* + 사설 동물 보호시설에 대한 운영기준 등이 부재함. 사설동물보호시설은 그 운영에 있어 ｢동물보호법｣의 적용 대상이 아니기 때문에 지방자치단체가 운영하는 직영 및 위탁형 동물보호센터에 적용이 되는 보호동물에 대한 인도적 처리, 즉 안락사의 대상, 원칙, 절차 및 방법에 대한 규정이 적용되지 않음.
  + 지난 2019년 3월 농림축산식품부에서 발표한 「사설 동물보호소 실태조사 및 관리 방안 마련 연구 최종보고서」에 따르면 전국 사설보호소 83곳 중 60곳은 ‘국토의 계획 및 이용에 관한 법률’, ‘수도법’, ‘수질 및 수생태계 보전에 관한 법률’ 등의 법률을 위반함.

※ 국내 사설보호소는 총 83개소로 보호하고 있는 동물의 수 총 14,697마리 (2018년 12월 기준)

※ 82개 사설보호소 중 73개 사설보호소(토지정보 파악이 어려운 9곳 제외) 토지정보를 분석한 결과 60개의 사설보호소가 최소 1개 이상의 토지 관련 법률을 위반하면서 운영 중임.

* + 현행 동물보호법 15조에 따르면 유기동물의 구조·보호에 관해 동물보호센터를 설치·운영할 수 있는 주체는 시·도지사 또는 시장·군수·구청장과 이들이 지정하는 기관이나 단체에 한정되어 지방자치단체와 무관한 민간인이 운영하는 사설보호소에 대한 규정은 전혀 없음.

3) 시사점

* + 일부 보호소에서 파보 바이러스 등의 감염병이 돌아 죽는 경우가 발생. 유실·유기동물의 경우 길에서의 생활로 감염병에 노출 가능성이 높은 반면 많은 보호소들이 예산과 공간 등의 문제로 새로 입소한 동물을 위한 격리공간을 갖추지 않아 이런 사고들이 반복될 가능성이 있음. 보호소 입소절차 및 위생관리에 대한 정비가 요구됨.
  + 따라서 사설 동물 보호시설 관련 제도 도입과 더불어 국가 운영 ‘공공 동물보호센터’의 추가적 설립 추진의 필요성을 시사함.

※ 참고자료: 동물 자유 연대 「2016-2020 유실유기동물 분석 보고서」, 유제범 (경제산업조사실 산업자원팀 입법조사관) <국내 동물보호시설의 운영 현황과 개선방향> 2019.03.06

### 유기동물 처리 현황

* + 유기동물 중 다른 가정으로 분양되는 동물의 수에 비해 자연사 및 안락사의 비율이 높음.

※ 보호소 내에서 죽음을 맞이하는 비율(자연사 + 안락사)은 2016년 47.3%, 2017년 49.2%, 2018 년 50.2%, 2019년 52.5%로 매년 증가

* + 일부 지자체의 경우 노킬(No-Kill) 정책을 펴고 있지만 다수의 지자체의 경우 일정 기간 경과 후 안락사 시행.
  + 입소 동물 중 21~25% 정도가 안락사(2016-2020) - 안락사 외에도 상해, 질병, 원인미상의 이유로 죽음에 이르는 자연사도 전체 입소 동물의 26~28%에 달함(2016-2020)

### 동물 보호 정책 추진

1) 동물 등록제 추진

* + 동물등록제는 대통령령에서 정하는 “등록 대상 동물”의 보호와 유기 방지, 유기 또는 유실된 동물 발견 시 소유자 확인 및 소유자의 책임의식 고취 등을 목적으로 도입됨.
  + 시∙도지사는 등록대상동물의 보호와 유실·유기방지, 유실·유기된 동물 발견 시 소유자로 하여금 시장∙군수 또는 구청장에게 소유하고 있는 “등록대상동물”을 등록하게 할 수 있음.
  + 등록 동물의 범위, 등록방법, 등록사항과 절차 그 밖에 등록에 관하여 필요한 사항은 농림축산식품부령이 정하는 것을 제외하고는 시∙도의 조례를 따름.

2) 서울시 길고양이 중성화 사업 추진

* + 서울시 각 자치구에서는 전문기관과 위탁계약을 체결하여 길고양이 중성화사업을 진행함.
  + 수술이 필요한 길고양이가 있다면 관할구청으로 길고양이 중성화(TNR)을 신청하고 신청된 길고양이는 포획 후 중성화 시술을 하고 다시 제자리로 방사함.

### 동물 복지 관련 논란 및 문제점

1) 길고양이 돌봄 논란 (‘캣맘’, ‘캣대디’ 논란)

* + 길고양이에게 밥을 주고 돌보는 이른바 ‘캣맘’과 ‘캣대디’는 사회적으로 논란의 대상이 되고 있음.
  + 불쌍한 동물을 보살피는 따뜻한 마음의 소유자라는 긍정적 평가가 있는 반면, 길고양이 개체수를 증가시켜 여러 부작용을 일으킨다는 부정적 평가가 공존함.

2) 고통사 논란

* + 지자체 동물보호센터에서 보호기간이 끝난 동물들 대부분이 ‘고통사’ 당했다는 조사가 발표됨.
  + 동물보호단체 비글구조네트워크에 따르면 안락사 위탁을 받은 동물병원 수의사는 다른 유기견들이 보는 앞에서 유기견을 안락사 해온 것으로 드러남. 이는 동물보호법 제8조 2항 ‘같은 종류의 다른 동물이 보는 앞에서 죽음에 이르는 행위 금지”를 어긴 것임.
  + 마취를 하지 않고 호흡 마비를 불러 일으키는 석시콜린(근육이완제)만으로 안락사를 하고 있는 곳도 발견됨. 이는 동물보호법 제 22조 3항 ‘동물의 안락사는 반드시 마취를 한 뒤 심장정지, 호흡마비를 유발하는 약제를 사용해야 한다.’ 를 어긴 것임.

## 주제설계를 위한 마인드맵



## 분석 상세코드 등

표 preprocessing.R

|  |
| --- |
| library(readxl)  rm(list=ls())  df <- read\_excel("./data/EDA\_FINAL.xlsx" , na = "NA")  name = df[,1]  df = df[,-1]  df = data.frame(df, row.names = name$구별)  head(df)  scale\_df = scale(df)  human = cbind(df[,1:7], df[,12:49])  head(human)  animal <- cbind(df[,8:11], df[,50:52], df[,55:66])  head(animal)  infra <- cbind(df[,53:54])  # 동물보호소는 제외 : 상관성이 너무 낮음  ########################################################################  # 상관행렬  show\_cor <- function(df.data) {    library(corrplot)    s\_data = scale(df.data, center = TRUE)    cor\_h = cor(s\_data)    corrplot(cor\_h, is.corr = FALSE, method = "square")  }  show\_cor(human)  discor\_human\_1 = cbind(human[,c(-3,-5,-7,-10,-11, -45)])  # 단독주택, 연립주택, 월소득, 기타, 농림어업 숙련 종사자, 100세 이상  show\_cor(discor\_human\_1)  result <- discor\_human\_1  home <- cbind(result[,3:4])  job <- cbind(result[,5:12])  gagu <- cbind(result[,13:19])  age <- cbind(result[,20:39])  show\_cor(animal)  result\_anim <- animal[, c(-5, -6, -7, -15, -16)]  show\_cor(result\_anim)  home\_anim <- cbind(result\_anim[,1:4])  state\_anim <- cbind(result\_anim[,5:11])  another\_anim <- cbind(result\_anim[,12:14])  ########################################################################  ########################################################################  # FA  library(psych)  library(GPArotation)  anal\_FA\_pre <- function(df.data) {    scale\_df <- scale(df.data, center = TRUE)    X\_cor <- cor(scale\_df, use="pairwise.complete.obs")    scree(X\_cor, factors = FALSE)    fa.parallel(scale\_df, fm = 'minres', fa = 'fa')  }  anal\_FA <- function(df.data, nf) {    scale\_df <- scale(df.data, center = TRUE)    fa.model <- fa(scale\_df, nfactors=nf, n.obs=N, rotate="varimax")    fa.diagram(fa.model)    print(fa.model$loadings, cutoff = 0)    return (fa.model$loadings)  }  # n인가구 : 2개  head(gagu)  anal\_FA\_pre(gagu)  w\_gagu <- anal\_FA(gagu, 2)  s\_gagu <- scale(gagu, center = TRUE)  m\_gagu <- matrix(s\_gagu, nrow=25)  gagu\_MR1 <- m\_gagu  %\*% w\_gagu[,1]  # 다인가구  gagu\_MR2 <- m\_gagu  %\*% w\_gagu[,2]  # 1인가구  avg\_gagu\_MR1 <- apply(s\_gagu[,2:7], 1, sum) / ncol(s\_gagu[,2:7])  # 다인가구  avg\_gagu\_MR2 <- s\_gagu[,1] # 1인가구  # 직업 : 2개  anal\_FA\_pre(job)  w\_job <- anal\_FA(job, 2)  s\_job <- scale(job, center = TRUE)  m\_job <- matrix(s\_job, nrow=25)  job\_MR1 <- m\_job  %\*% w\_job[,1]  # 단순노무종사자, 기능원 및 관련기능종사자, 장치기계조작 및 조립종사자, 판매종사자, 서비스종사자  job\_MR2 <- m\_job  %\*% w\_job[,2]  # 전문가 및 관련 종사자, 관리자, 사무종사자  avg\_job\_MR1 <- apply(s\_job[,c(-1,-4,-7)],1,sum) / ncol(s\_job[,c(-1,-4,-7)])  avg\_job\_MR2 <- apply(s\_job[,c(1,4,7)],1,sum) / ncol(s\_job[,c(1,4,7)])  # 연령 : 3개  anal\_FA\_pre(age)  w\_age <- anal\_FA(age, 3)  s\_age <- scale(age, center = TRUE)  m\_age <- matrix(s\_age, nrow=25)  age\_MR1 <- m\_age  %\*% w\_age[,1]  # 그 이외  age\_MR2 <- m\_age  %\*% w\_age[,2]  # , 60~80 소비력있는 고령층  age\_MR3 <- m\_age  %\*% w\_age[,3]  # 20~25 , 25~40 소비력있는 젊은층  avg\_age\_MR1 <- apply(s\_age[, -c(5:8, 13:18)],1,sum) / ncol(s\_age[, -c(5:8, 13:18)])  avg\_age\_MR2 <- apply(s\_age[,c(13:18)],1,sum) / ncol(s\_age[,c(13:18)])  # 60 ~ 89  avg\_age\_MR3 <- apply(s\_age[,c(5:8)],1,sum) / ncol(s\_age[,c(5:8)])   # 20 ~ 39  # 주택 : 1개  anal\_FA\_pre(home)  w\_home <- anal\_FA(home, 1)  s\_home <- scale(home, center = TRUE)  m\_home <- matrix(s\_home, nrow=25)  home\_MR1 <- m\_home  %\*% w\_home[,1]  # 아파트, 다세대 주택  avg\_home\_MR1 <- apply(s\_home,1,sum) / ncol(s\_home)  # 반려동물 주택 수  anal\_FA\_pre(home\_anim)  w\_home\_anim <- anal\_FA(home\_anim, 2)  s\_home\_anim <- scale(home\_anim, center = TRUE)  m\_home\_anim <- matrix(s\_home\_anim, nrow=25)  home\_anim\_MR1 <- m\_home\_anim  %\*% w\_home\_anim[,1]  # 다세대, 연립 주택 / 주택 형태에 사람이 다수 사는 곳  home\_anim\_MR2 <- m\_home\_anim  %\*% w\_home\_anim[,2]  # 아파트, 단독주택  avg\_home\_anim\_MR1 <- apply(s\_home\_anim[, -c(1:2)],1,sum) / ncol(s\_home\_anim[, -c(1:2)])  avg\_home\_anim\_MR2 <- apply(s\_home\_anim[, c(1:2)],1,sum) / ncol(s\_home\_anim[, c(1:2)])  # 반려동물 보호 현황  anal\_FA\_pre(state\_anim)  w\_state\_anim <- anal\_FA(state\_anim, 2)  s\_state\_anim <- scale(state\_anim, center = TRUE)  m\_state\_anim <- matrix(s\_state\_anim, nrow=25)  state\_anim\_MR1 <- m\_state\_anim  %\*% w\_state\_anim[,1]  # 입양, 반환, 안락사 / 사람의 적극 개입  state\_anim\_MR2 <- m\_state\_anim  %\*% w\_state\_anim[,2]  # 방사, 자연사, 보호중  / 사람의 소극 개입 및 방치  avg\_state\_anim\_MR1 <- apply(s\_state\_anim[, -c(1,5,6,7)],1,sum) / ncol(s\_state\_anim[, -c(1,5,6,7)])  # 기증은 제외  avg\_state\_anim\_MR2 <- apply(s\_state\_anim[, c(1, 6,7)],1,sum) / ncol(s\_state\_anim[, c(1, 6,7)])    # 방사, 자연사, 보호중  # 그 외  anal\_FA\_pre(another\_anim)  w\_another\_anim <- anal\_FA(another\_anim, 1)  s\_another\_anim <- scale(another\_anim, center = TRUE)  m\_another\_anim <- matrix(s\_another\_anim, nrow=25)  another\_anim\_MR1 <- m\_another\_anim  %\*% w\_another\_anim[,1] # 동물 등록수, 반령동물 인구, 총 유기동물수  avg\_another\_anim\_MR1 <- apply(s\_another\_anim[, c(1:3)],1,sum) / ncol(s\_another\_anim[, c(1:3)])  # 기증은 제외  new\_df <- cbind(df[,1:2],                  gagu\_MR1, gagu\_MR2,                  job\_MR1, job\_MR2,                  age\_MR1, age\_MR2, age\_MR3,                  home\_MR1,                  home\_anim\_MR1, home\_anim\_MR2,                  state\_anim\_MR1, state\_anim\_MR2,                  another\_anim\_MR1)  new\_df\_human <- cbind(df[,1:2],                  gagu\_MR1, gagu\_MR2,                  job\_MR1, job\_MR2,                  age\_MR1, age\_MR2, age\_MR3,                  home\_MR1)  new\_df\_anim <- cbind(home\_anim\_MR1, home\_anim\_MR2,                  state\_anim\_MR1, state\_anim\_MR2,                  another\_anim\_MR1)  avg\_new\_df <- cbind(df[,1:2],                  avg\_gagu\_MR1, avg\_gagu\_MR2,                  avg\_job\_MR1, avg\_job\_MR2,                  avg\_age\_MR1, avg\_age\_MR2, avg\_age\_MR3,                  avg\_home\_MR1,                  avg\_home\_anim\_MR1, avg\_home\_anim\_MR2,                  avg\_state\_anim\_MR1, avg\_state\_anim\_MR2,                  avg\_another\_anim\_MR1)  avg\_new\_df\_human <- cbind(df[,1:2],                  avg\_gagu\_MR1, avg\_gagu\_MR2,                  avg\_job\_MR1, avg\_job\_MR2,                  avg\_age\_MR1, avg\_age\_MR2, avg\_age\_MR3,                  avg\_home\_MR1)  avg\_new\_df\_anim <- cbind(avg\_home\_anim\_MR1, avg\_home\_anim\_MR2,                  avg\_state\_anim\_MR1, avg\_state\_anim\_MR2,                  avg\_another\_anim\_MR1)  anal\_FA\_pre(new\_df)  w\_new\_df <- anal\_FA(new\_df, 3)  s\_new\_df <- scale(new\_df, center = TRUE)  m\_new\_df <- matrix(s\_new\_df, nrow=25)  new\_df\_MR1 <- m\_new\_df  %\*% w\_new\_df[,1]  new\_df\_MR2 <- m\_new\_df  %\*% w\_new\_df[,2]  new\_df\_MR3 <- m\_new\_df  %\*% w\_new\_df[,3]  anal\_FA\_pre(new\_df\_human)  w\_new\_df\_human <- anal\_FA(new\_df\_human, 1)  s\_new\_df\_human <- scale(new\_df\_human, center = TRUE)  m\_new\_df\_human <- matrix(s\_new\_df\_human, nrow=25)  new\_df\_human\_MR1 <- m\_new\_df\_human  %\*% w\_new\_df\_human[,1]  anal\_FA\_pre(new\_df\_anim)  w\_new\_df\_anim <- anal\_FA(new\_df\_anim, 3)  s\_new\_df\_anim <- scale(new\_df\_anim, center = TRUE)  m\_new\_df\_anim <- matrix(s\_new\_df\_anim, nrow=25)  new\_df\_anim\_MR1 <- m\_new\_df\_anim  %\*% w\_new\_df\_anim[,1]  new\_df\_anim\_MR2 <- m\_new\_df\_anim  %\*% w\_new\_df\_anim[,2]  new\_df\_anim\_MR3 <- m\_new\_df\_anim  %\*% w\_new\_df\_anim[,3]  anal\_FA\_pre(avg\_new\_df)  w\_avg\_new\_df <- anal\_FA(avg\_new\_df, 3)  s\_avg\_new\_df <- scale(avg\_new\_df, center = TRUE)  m\_avg\_new\_df <- matrix(s\_avg\_new\_df, nrow=25)  avg\_new\_df\_MR1 <- m\_avg\_new\_df  %\*% w\_avg\_new\_df[,1]  avg\_new\_df\_MR2 <- m\_avg\_new\_df  %\*% w\_avg\_new\_df[,2]  avg\_new\_df\_MR3 <- m\_avg\_new\_df  %\*% w\_avg\_new\_df[,3]  anal\_FA\_pre(avg\_new\_df\_human)  w\_avg\_new\_df\_human <- anal\_FA(avg\_new\_df\_human, 1)  s\_avg\_new\_df\_human <- scale(avg\_new\_df\_human, center = TRUE)  m\_avg\_new\_df\_human <- matrix(s\_avg\_new\_df\_human, nrow=25)  avg\_new\_df\_human\_MR1 <- m\_avg\_new\_df\_human  %\*% w\_avg\_new\_df\_human[,1]  anal\_FA\_pre(avg\_new\_df\_anim)  w\_avg\_new\_df\_anim <- anal\_FA(avg\_new\_df\_anim, 3)  s\_avg\_new\_df\_anim <- scale(avg\_new\_df\_anim, center = TRUE)  m\_avg\_new\_df\_anim <- matrix(s\_avg\_new\_df\_anim, nrow=25)  avg\_new\_df\_anim\_MR1 <- m\_avg\_new\_df\_anim  %\*% w\_avg\_new\_df\_anim[,1]  avg\_new\_df\_anim\_MR2 <- m\_avg\_new\_df\_anim  %\*% w\_avg\_new\_df\_anim[,2]  avg\_new\_df\_anim\_MR3 <- m\_avg\_new\_df\_anim  %\*% w\_avg\_new\_df\_anim[,3]  infra <- infra[,2]  colnames(infra) <- c("infra")  head(infra)  region = c('강남구', '강동구','강북구','강서구','관악구','광진구',             '구로구','금천구','노원구','도봉구','동대문구','동작구',             '마포구','서대문구','서초구','성동구','성북구','송파구',             '양천구','영등포구','용산구','은평구','종로구','중구','중랑구')  exportDF <- data.frame(new\_df\_MR1, new\_df\_MR2, new\_df\_MR3, infra,        new\_df\_human\_MR1, new\_df\_anim\_MR1, new\_df\_anim\_MR2, new\_df\_anim\_MR3, infra,        avg\_new\_df\_MR1, avg\_new\_df\_MR2, avg\_new\_df\_MR3, infra,        avg\_new\_df\_human\_MR1,avg\_new\_df\_anim\_MR1, avg\_new\_df\_anim\_MR2, avg\_new\_df\_anim\_MR3, infra        )  rownames(exportDF) <- region  head(exportDF)  library(openxlsx)  write.xlsx(exportDF, sheetName = "sheet1", file = "./data/FA\_RESULT.xlsx")  ########################################################################  ########################################################################  # PCA  anal\_pca <- function(df.data) {    library(factoextra)    pca\_data <- prcomp(df.data, center = T, scale. = T)    screeplot(pca\_data, main="", col="blue", type="lines", pch=3, npcs = length(pca\_data$sdev))    fviz\_pca\_ind(pca\_data, col.ind="cos2")    biplot(pca\_data)    print(summary(pca\_data))      return(pca\_data$rotation)  }  w\_pca <- anal\_pca(new\_df)  s\_pca <-scale(new\_df, center=TRUE)  m\_pca <- matrix(s\_pca, nrow=25)  df\_PC1 <- m\_pca  %\*% w\_pca[,1]  df\_PC2 <- m\_pca  %\*% w\_pca[,2]  df\_PC3 <- m\_pca  %\*% w\_pca[,3]  df\_PC4 <- m\_pca  %\*% w\_pca[,4]  w\_pca\_avg <- anal\_pca(avg\_new\_df)  s\_pca\_avg <-scale(avg\_new\_df, center=TRUE)  m\_pca\_avg <- matrix(s\_pca\_avg, nrow=25)  df\_avg\_PC1 <- m\_pca\_avg  %\*% w\_pca\_avg[,1]  df\_avg\_PC2 <- m\_pca\_avg  %\*% w\_pca\_avg[,2]    w\_pca\_human <- anal\_pca(new\_df\_human)  s\_pca\_human <-scale(new\_df\_human, center=TRUE)  m\_pca\_human <- matrix(s\_pca\_human, nrow=25)  human\_PC1 <- m\_pca\_human  %\*% w\_pca\_human[,1]  human\_PC2 <- m\_pca\_human  %\*% w\_pca\_human[,2]  w\_pca\_anim <- anal\_pca(new\_df\_anim)  s\_pca\_anim <-scale(new\_df\_anim, center=TRUE)  m\_pca\_anim <- matrix(s\_pca\_anim, nrow=25)  anim\_PC1 <- m\_pca\_anim  %\*% w\_pca\_anim[,1]  anim\_PC2 <- m\_pca\_anim  %\*% w\_pca\_anim[,2]  anim\_PC3 <- m\_pca\_anim  %\*% w\_pca\_anim[,3]  anim\_PC4 <- m\_pca\_anim  %\*% w\_pca\_anim[,4]  w\_pca\_avg\_human <- anal\_pca(avg\_new\_df\_human)  s\_pca\_avg\_human <-scale(avg\_new\_df\_human, center=TRUE)  m\_pca\_avg\_human <- matrix(s\_pca\_avg\_human, nrow=25)  avg\_human\_PC1 <- m\_pca\_avg\_human  %\*% w\_pca\_avg\_human[,1]  avg\_human\_PC2 <- m\_pca\_avg\_human  %\*% w\_pca\_avg\_human[,2]  w\_pca\_avg\_anim <- anal\_pca(avg\_new\_df\_anim)  s\_pca\_avg\_anim <-scale(avg\_new\_df\_anim, center=TRUE)  m\_pca\_avg\_anim <- matrix(s\_pca\_avg\_anim, nrow=25)  avg\_anim\_PC1 <- m\_pca\_avg\_anim  %\*% w\_pca\_avg\_anim[,1]  avg\_anim\_PC2 <- m\_pca\_avg\_anim  %\*% w\_pca\_avg\_anim[,2]  export\_PCA\_DF <- data.frame(df\_PC1, df\_PC2, df\_PC3, df\_PC4, infra,                              human\_PC1, human\_PC2, anim\_PC1, anim\_PC2, anim\_PC3, anim\_PC4, infra,                              df\_avg\_PC1, df\_avg\_PC2, infra,                              avg\_human\_PC1, avg\_human\_PC2, avg\_anim\_PC1, avg\_anim\_PC2, infra  )  rownames(export\_PCA\_DF) <- region  head(export\_PCA\_DF)  library(openxlsx)  write.xlsx(export\_PCA\_DF, sheetName = "sheet1", file = "./data/PCA\_RESULT.xlsx")  ######################################################################## |

표 AHP\_Result.R

|  |
| --- |
| library(readxl)  rm(list=ls())  FA\_Result <- read\_excel("./data/FA\_RESULT.xlsx" , na = "NA")  weight <- c(0.153145333426747              , 0.140775127848327              , 0.270466021730839              , 0.227260323615609              , 0.0629030982838324)  weight  FA\_B <- FA\_Result[,5:9]  FA\_D <- FA\_Result[,14:18]  s\_FA\_B <- scale(FA\_B, center = TRUE)  s\_FA\_D <- scale(FA\_D, center = TRUE)  s\_FA\_B \* weight  sw\_FA\_B <- s\_FA\_B \* weight  r\_FA\_B <-apply(sw\_FA\_B, 1, sum)   # result\_FA\_B  r\_FA\_B  sw\_FA\_D <- s\_FA\_D \* weight  r\_FA\_D <-apply(sw\_FA\_D, 1, sum)  # result\_FA\_D  r\_FA\_D  region = c('강남구', '강동구','강북구','강서구','관악구','광진구',             '구로구','금천구','노원구','도봉구','동대문구','동작구',             '마포구','서대문구','서초구','성동구','성북구','송파구',             '양천구','영등포구','용산구','은평구','종로구','중구','중랑구')  score\_AHP\_B <- r\_FA\_B  score\_AHP\_D <- r\_FA\_D  AHP\_RESULT <- data.frame(cbind(score\_AHP\_B, score\_AHP\_D))  rownames(AHP\_RESULT) <- region  head(AHP\_RESULT)  write.csv(AHP\_RESULT,file = "./data/AHP\_RESULT.csv") |

표 Clustering\_Analysis.ipynb

|  |
| --- |
| #!/usr/bin/env python  # coding: utf-8  # In[25]:  from sklearn.cluster import KMeans  import matplotlib.pyplot as plt  import numpy as np  import pandas as pd  # get\_ipython().run\_line\_magic('matplotlib', 'inline')  # In[26]:  ydf = pd.read\_excel('../data/PCA\_RESULT.xlsx')  # In[27]:  ydf  # In[30]:  #스케일링  from sklearn.preprocessing import StandardScaler  scaler = StandardScaler()  scaler.fit\_transform(l\_df)  sl\_df = scaler.fit\_transform(l\_df)  scaler.fit\_transform(a\_df)  sa\_df = scaler.fit\_transform(a\_df)  # In[31]:  # GMM  from sklearn.mixture import GaussianMixture  # In[32]:  # n\_components로 군집 개수 설정  n=4  GMM = GaussianMixture(n\_components=n, random\_state=0, )#covariance\_type = 'full')  sl\_labels = GMM.fit\_predict(sl\_df)  sa\_labels = GMM.fit\_predict(sa\_df)  GMM\_l\_CL = pd.DataFrame(sl\_labels)  GMM\_a\_CL = pd.DataFrame(sa\_labels)  gm\_sl\_df = pd.DataFrame(sl\_df)  gm\_sl\_df['sl\_labels'] = sl\_labels  gm\_sl\_df.columns = ["ftr1", 'ftr2', 'ftr3', 'ftr4', 'ftr5', 'sl\_labels']  gm\_sl\_df  def visualize\_cluster\_plot(clusterobj, dataframe, label\_name, iscenter=True):      if iscenter :          centers = clusterobj.cluster\_centers\_      unique\_labels = np.unique(dataframe[label\_name].values)      markers=['o', 's', '^', 'x', '\*']      isNoise=False      for label in unique\_labels:          label\_cluster = dataframe[dataframe[label\_name]==label]          if label == -1:              cluster\_legend = 'Noise'              isNoise=True          else :              cluster\_legend = 'Cluster '+str(label)          plt.scatter(x=label\_cluster['ftr1'], y=label\_cluster['ftr2'], s=70,\                      edgecolor='k', marker=markers[label], label=cluster\_legend)            if iscenter:              center\_x\_y = centers[label]              plt.scatter(x=center\_x\_y[0], y=center\_x\_y[1], s=250, color='white',                          alpha=0.9, edgecolor='k', marker=markers[label])              plt.scatter(x=center\_x\_y[0], y=center\_x\_y[1], s=70, color='k',\                          edgecolor='k', marker='$%d$' % label)      if isNoise:          legend\_loc='upper center'      else: legend\_loc='upper right'        plt.legend(loc=legend\_loc)      # plt.savefig('../graph/CL\_gmm.png')      plt.show()      visualize\_cluster\_plot(GMM,gm\_sl\_df, 'sl\_labels', iscenter = False )  from scipy.cluster.hierarchy import linkage, dendrogram  # In[17]:  hier\_l\_CL = linkage(sl\_df,method='complete')  hier\_a\_CL = linkage(sa\_df,method='complete')  plt.figure(figsize=(40,20))  dendrogram(hier\_l\_CL,               leaf\_rotation=90,             leaf\_font\_size=20,  )  plt.show()  # plt.savefig('hira\_fa\_a.png',dpi=80)  from sklearn.cluster import MeanShift, estimate\_bandwidth  gu={'gu':['강남구','강동구','강북구','강서구','관악구',  '광진구','구로구','금천구','노원구','도봉구',  '동대문구','동작구','마포구','서대문구','서초구',  '성동구','성북구','송파구','양천구','영등포구',  '용산구','은평구','종로구','중구','중랑구'  ]}  clust\_gu = pd.DataFrame(gu)  ms\_l\_CL = sl\_df  ms\_a\_CL = sa\_df  names = ['ms\_l\_CL','ms\_a\_CL']  for i in names:      bandwidth=estimate\_bandwidth(eval(i))      meanshift=MeanShift(bandwidth=bandwidth)      labels=meanshift.fit\_predict(eval(i))      clust\_gu[i]=labels  clust\_gu  # In[33]:  columnNames=['gu','MS\_I\_CL','MS\_a\_CL', 'GMM\_l\_CL', 'GMM\_a\_CL']  final = pd.DataFrame(np.hstack((clust\_gu, GMM\_l\_CL,GMM\_a\_CL )), columns=columnNames)  final.to\_csv('PCA\_AC\_CL.csv', sep=',', na\_rep='NaN', encoding = 'cp949')  FA\_A = t\_df.iloc[:,12:17]  FA\_C = t\_df.iloc[:,18:20]  for i in range(2,12,2):      print(silhouette\_score(FA\_A,t\_df.iloc[:,i].values))   # meanshift, GMM, HIER, kmeans, dbscan  for i in range(3,12,2):      print(silhouette\_score(FA\_C,t\_df.iloc[:,i].values))  # meanshift, GMM, HIER, kmeans, dbscan |

표 Clustering\_Anlaysis.R

|  |
| --- |
| library(readxl)  rm(list=ls())  df = as.data.frame(read\_excel("./data/PCA\_RESULT.xlsx"))    PCA\_A <- df[,1:5]  PCA\_C <- df[,13:14]  s\_PCA\_A <- scale(PCA\_A, center = TRUE)  s\_PCA\_C <- scale(PCA\_C, center = TRUE)  region = c('강남구', '강동구','강북구','강서구','관악구','광진구',             '구로구','금천구','노원구','도봉구','동대문구','동작구',             '마포구','서대문구','서초구','성동구','성북구','송파구',             '양천구','영등포구','용산구','은평구','종로구','중구','중랑구')  # rownames(s\_PCA\_A) <- region  # rownames(s\_PCA\_C) <- region  A\_kmeans<-kmeans(s\_PCA\_A,center=4)  km\_l\_CL <- A\_kmeans$cluster  C\_kmeans<- kmeans(s\_PCA\_C,center=4)  km\_a\_CL <- C\_kmeans$cluster  wss <- 0  for(i in 1:15) wss[i]<-sum(kmeans(s\_PCA\_A,centers = i)$withinss)  plot(1:15, wss, type="b",xlab = "Number of Clusters", ylab = "Within group sum of squares",main='FA\_A')  ######## dbscan  library(fpc)  show\_dbscan <- function(datadf, ep, mp) {    db<-dbscan(datadf,eps=ep,MinPts=mp)    str(db)    sum(db$isseed)    db$cluster  #  return(db$cluster)  }  db\_l\_CL <- show\_dbscan(s\_PCA\_A, 1.61, 2)  db\_l\_CL  db\_a\_CL <- show\_dbscan(s\_PCA\_C, 0.6, 2)  db\_a\_CL    final <- data.frame(cbind(km\_l\_CL, km\_a\_CL, db\_l\_CL, db\_a\_CL), row.names = region)  write.csv(x=final,file='./data/final\_CL.csv') |

표 Visuallize.R

|  |
| --- |
| #########################################################################################################  # Data Load  rm(list=ls())  library(readxl)  df\_1 <- read\_excel('./data/FA\_B\_MEANSHIFT.xlsx')  df\_1.data <- df\_1[,3:7]  df\_1.cluster <- df\_1[,2]  df\_2 <- read\_excel('./data/FA\_A\_KMEANS.xlsx')  df\_2.data <- df\_2[, 3:6]  df\_2.cluster <- df\_2[, 2]  df\_3 <- read\_excel('./data/FA\_A\_HIR.xlsx')  df\_3.data <- df\_3[, 3:6]  df\_3.cluster <- df\_3[, 2]  df\_4 <- read\_excel('./data/PCA\_B\_KMEANS.xlsx')  df\_4.data <- df\_4[, 3:7]  df\_4.cluster <- df\_4[, 2]  df\_5 <- read\_excel('./data/FA\_B\_HIR.xlsx')  df\_5.data <- df\_5[, 3:7]  df\_5.cluster <- df\_5[, 2]  #########################################################################################################  #########################################################################################################  # show silhouette  library(factoextra)  show\_sil <- function(clus, raw\_data) {    s\_df <- scale(raw\_data, center = TRUE)    clust <- as.integer(unlist(c(clus)))    sil <- silhouette(as.integer(unlist(clust)), dist(s\_df))    fviz\_silhouette(sil)  }  show\_sil(df\_1.cluster, df\_1.data) # meanshift, FA\_B  show\_sil(df\_2.cluster, df\_2.data) # kmeans, FA\_A  show\_sil(df\_3.cluster, df\_3.data) # hiera, FA\_A  show\_sil(df\_4.cluster, df\_4.data) # kmeans, PCA\_B  show\_sil(df\_5.cluster, df\_5.data) # hiera, FA\_B  #########################################################################################################  #########################################################################################################  # show visual clustering  library(cluster)  library(factoextra)  show\_viz\_kmeans <- function(raw\_data, nc, seed\_num) {    # nc는 군집 개수    set.seed(seed\_num)    s\_df <- scale(raw\_data, center = TRUE)    km <-kmeans(s\_df, nc)    fviz\_cluster(km, data = s\_df, stand=F, geom = "point" )  }  show\_viz\_dend <- function(raw\_data, nc) {    # nc는 군집 개수    s\_df <- scale(raw\_data, center = TRUE)    hk <-hkmeans(s\_df, nc)    fviz\_dend(hk, cex = 0.6, palette = "jco",              rect = TRUE, rect\_border = "jco", rect\_fill = TRUE)  }  #########################################################################################################  #########################################################################################################  # graph  s\_df <- scale(df\_1.data)  anal\_pca <- function(df.data) {    library(factoextra)    pca\_data <- prcomp(df.data, center = T, scale. = T)    screeplot(pca\_data, main="", col="blue", type="lines", pch=3, npcs = length(pca\_data$sdev))    fviz\_pca\_ind(pca\_data, col.ind="cos2")    # biplot(pca\_data)    print(summary(pca\_data))    return(pca\_data$rotation)  }    library(ggplot2)  test <- data.frame(df\_PC1, df\_PC2, df\_1.cluster)  test  test\_plot <- ggplot(data = test, aes(x = test$df\_PC1, y = test$df\_PC2, group = test$FA\_B\_MEANSHIFT  color = test$FA\_B\_MEANSHIFT)) +    geom\_point(shape = 19, size = 4)  test\_plot\_2 <- test\_plot + annotate("rect", xmin=0, xmax=2.6, ymin=0, ymax=0.8, alpha=0.1, fill="red")    test\_plot\_2  #########################################################################################################  # graph로 적당한 scatter 변수 찾기  panel.fun <- function(x, y, ...) {    # 출처: https://rfriend.tistory.com/228 [R, Python 분석과 프로그래밍의 친구 (by R Friend)]    horizontal <- (par("usr")[1] + par("usr")[2]) / 2;    vertical <- (par("usr")[3] + par("usr")[4]) / 2;    text(horizontal, vertical, format(abs(cor(x,y)), digits=2))  }  plot(df\_1.data, pch = 8, bg = c("red", "green3", "blue", "black"),       upper.panel = panel.fun,col=1:4, cex = 2)  #########################################################################################################  # meanshift  show\_pca <- function(raw\_data) {    w\_pca <- anal\_pca(raw\_data)    s\_pca <-scale(raw\_data, center=TRUE)    m\_pca <- matrix(s\_pca, nrow=25)    df\_PC1 <- m\_pca  %\*% w\_pca[,1]    df\_PC2 <- m\_pca  %\*% w\_pca[,2]    # df\_PC3 <- m\_pca  %\*% w\_pca[,3]      print(df\_PC1)    print(df\_PC2)    # print(df\_PC3)      plot(df\_PC1,df\_PC2, pch = 8, col = 1:4, cex = 2)    legend("topright", legend = c("cluster 0", "cluster 1", "cluster 2", "cluster 3"), fill=c("blue", "red" , "green", "black"))  }  # meanshift, FA\_B  show\_pca(df\_1.data)  # kmeans  show\_viz\_kmeans <- function(raw\_data, nc, seed\_num) {    # nc는 군집 개수    set.seed(seed\_num)    s\_df <- scale(raw\_data, center = TRUE)    km <-kmeans(s\_df, nc)    fviz\_cluster(km, data = s\_df, stand=F, ellipse = T )  }  # kmeans, FA\_A  show\_viz\_kmeans(df\_2.data, 4, 178)  # show\_pca(df\_2.data)  # hiera, FA\_A  show\_viz\_dend(df\_3.data, 4)  show\_pca(df\_3.data)  # kmeans, PCA\_B  show\_viz\_kmeans(df\_4.data, 4, 750)  show\_pca(df\_4.data)  # hiera, FA\_B  show\_viz\_dend(df\_5.data, 4)  show\_pca(df\_5.data)  #########################################################################################################  #########################################################################################################  # dbScan visualize  df = as.data.frame(read\_excel("./data/PCA\_RESULT.xlsx"))  PCA\_A <- df[,1:5]  PCA\_C <- df[,13:14]  s\_PCA\_A <- scale(PCA\_A, center = TRUE)  s\_PCA\_C <- scale(PCA\_C, center = TRUE)  library(fpc)  show\_dbscan <- function(datadf, ep, mp) {    db<-dbscan(datadf,eps=ep,MinPts=mp)    str(db)    sum(db$isseed)    db$cluster    #  return(db$cluster)  }  show\_pca(PCA\_A)  db\_visual <-  dbscan(s\_PCA\_C,eps=0.6,MinPts=2)  db\_visual  fviz\_cluster(db\_visual, s\_PCA\_C, stand = F, frame = F, ellipse = T, geom = "point")  ######################################################################################################### |

표 countif.ipynb

|  |
| --- |
| import pandas as pd  import xml.etree.ElementTree as et  FILE\_PATH = '/content/drive/MyDrive/Colab Notebooks/Test/'  df = pd.read\_csv(FILE\_PATH + '교차\_아파트.csv', encoding='cp949')  df.head  test\_df = df.iloc[0:500, :]  test\_df.head  new\_df = df.iloc[:,0:2]  new\_df.head  new\_df['count'] = new\_df.groupby(['id'])['left'].transform('count')  export\_df = new\_df.drop\_duplicates(['id'])  export\_df = export\_df.loc[:,['id', 'count']]  export\_df |

표 grid\_processing.R

|  |
| --- |
| rm(list=ls())  df <- read.csv("./data/최종\_그리드.csv")  df[is.na(df)] <- 0  X <- df[, c(6:10, 12)]  s\_X <- scale(X, center = TRUE)  result <- apply(s\_X, 1, sum)  export\_df <- cbind(df[,1], result, scale(result, center = TRUE))    colnames(export\_df) <- c("id", "Score", "ScaleScore")  write.csv(x=export\_df,file='./data/all\_score.csv')  Score <- result  ScaleScore <- scale(result, center = TRUE)  report <- cbind(df[,c(1:10, 12)], Score, ScaleScore)  head(report)  write.csv(x=report,file='./data/all\_score\_for\_report.csv') |

1. 이는 통계청의 2019년 인구총조사, 농림축산식품부 동물등록정보 현황, 전국 20세 이상 남녀 1,000명을 대상으로 한 설문조사 결과를 통해 추정한 수치이다. [↑](#footnote-ref-2)
2. 보호소 내에서 죽음을 맞이하는 비율(자연사 + 안락사)은 2016년 47.3%, 2017년 49.2%, 2018 년 50.2%, 2019년 52.5%로 매년 증가 [↑](#footnote-ref-3)
3. 민간위탁이 228개소로 가장 많음 [↑](#footnote-ref-4)
4. 서울시는 동물 복지 센터를 2023년까지 권역별로 현재 1개소에서 4개소로 조성하도록 지시함 [↑](#footnote-ref-5)
5. 서울시의 동물등록 비율 18.9%, 동물 보호센터 비율 9.3% 로 보호센터의 비율이 현저히 낮음 [↑](#footnote-ref-6)