**T.C.**

**KARADENİZ TEKNİK ÜNİVERSİTESİ \* FEN FAKÜLTESİ**

**İSTATİSTİK ve BİLGİSAYAR BİLİMLERİ BÖLÜMÜ**

**IST4000 BİTİRME ÇALIŞMASI**

**MiniZinc İLE OPTİMİZASYON**

**Hazırlayanlar: 357795 Samet KARASU**

**348877 Ertuğrul EŞOL**

**Trabzon-2020**

**T.C.**

**KARADENİZ TEKNİK ÜNİVERSİTESİ \* FEN FAKÜLTESİ**

**İSTATİSTİK ve BİLGİSAYAR BİLİMLERİ BÖLÜMÜ**

**IST4000 BİTİRME ÇALIŞMASI**

**MiniZinc İLE OPTİMİZASYON**

**Hazırlayanlar: 357795 Samet KARASU**

**348877 Ertuğrul EŞOL**

**Tez Danışmanı: Dr. Öğr. Üyesi Tolga BERBER**

**Trabzon-2020**

# İÇİNDEKİLER

İÇİNDEKİLER II

GİRİŞ 1

**BÖLÜM I**

**1. OPTİMİZASYON’A GİRİŞ VE TEMEL KAVRAMLAR** 2

1.1. Optimizasyon Nedir ? 2

1.2. Doğrusal Programlama 4

1.2.1. Bir Doğrusal Programlama Modelinin Genel Yapısı 4

1.2.2. Doğrusal Programlamada Çözüm Yöntemleri 5

1.2.2.1. Grafik Yöntemi ve Örnekler 5

BÖLÜM II

**2. MiniZinc ile OPTİMİZASYON** 15

2.1.MiniZinc Nedir ? 15

2.1.1 MinZinc’in Özellikleri 15

2.1.2 MiniZinc’de Kullanılan Bazı Komutlar 16

2.1.3 Bir Optimizasyon Modeline Genel Bakış 17

2.2. Kurulum 17

2.3. MiniZinc ile İlk Adımlar 19

2.3.1. MiniZinc IDE 19

BÖLÜM III

**3. MiniZinc İLE UYGULAMALAR** 25

3.1. Optimizasyon Problemleri MiniZinc’e Aktarılması ve Çözümü 25

II

# GİRİŞ

Optimizasyon bir karar verme problemi için eniyi kararların belirlenmesine ilişkin sistematik yöntemleri içerir. Üzerinde durulan problem için anlamlı performans kriterlerinin ve dikkat edilmesi gereken kısıtların matematiksel olarak ifade edilerek formülasyonlarının geliştirilmesi optimizasyonun temelini oluşturur. Kararlar belirlenirken, mevcut kısıtları gözeterek eniyilik kriterine ilişkin bir amaç fonksiyonunun enküçüklenmesi veya enbüyüklenmesi söz konusudur.

MiniZinc ise ücretsiz ve açık kaynaklı bir kısıt modelleme dilidir ve  MiniZinc'i, önceden tanımlanmış kısıtlamalardan oluşan geniş bir kütüphaneden yararlanarak optimizasyon sorunlarını üst düzey, çözücüden bağımsız bir şekilde modellemek için kullanılmaktadır.

Sınırlı kaynakların etkin bir şekilde kullanılıp artan rekabet ortamında başarılı sonuçların elde edilebilmesi için optimizasyon tekniklerinin kullanılmasının önemi aşikardır ve bu nedenle de optimizasyonun çok çeşitli sektörde kullanılması kaçınılmazdır. Bu çalışmamızda söz konusu sektörler için bu denli önemli olan optimizasyon problemlerinin daha hızlı ve etkin olarak çözüme kavuşması için açık kaynaklı program olan MiniZinc ile optimizyon problemlerinin çözülmesi ve MiniZinc’in tanıtılması amaçlanmıştır.

# BİRİNCİ BÖLÜM

## 1. OPTİMİZASYON’A GİRİŞ VE TEMEL KAVRAMLAR

### **1.1. Optimizasyon Nedir?**

Optimizasyon, verilen amaç veya amaçlar doğrultusunda belirli kısıtlamaların sağlanarak en uygun çözümün elde edilme sürecidir. Optimizasyon, karar verme süreçlerini hızlandırmakta ve karar kalitesini arttırmakta kullanılarak gerçek hayatta karşılaşılan problemlerin etkin, doğru ve gerçek zamanlı çözümünde yararlanılmaktadır. Optimizasyon, ekonomik açılardan getirdiği kazançların yanında müşteri, işveren ve çalışanların tercih ve kısıtlarının karar sürecinde yer almasında ve sistemde yer alan kaynakların kalitesinin yükseltilmesinde de etkin bir şekilde başvurulan bir yöntem olarak kullanılmaktadır.

Matematiksel olarak ifade edilirse; optimizasyon, kısaca bir fonksiyonun minimize veya maksimize edilmesi olarak tanımlanabilir. Türkçe anlam karşılığı ise “uyumlaşma”, “uyumlaşım” ya da “uyumlaştırma” şeklindedir. Yani bir gerçel fonksiyonu minimize ya da maksimize etmek amacı ile gerçek ya da tamsayı değerlerini tanımlı bir aralıkta seçip fonksiyona yerleştirerek sistematik olarak bir problemi incelemek ya da çözmek işlemlerini ifade eder.

Optimizasyon, en iyi olanı aramaktır. Bulunan en iyi ise “optimum” olarak adlandırılır.

Optimizasyonun amacı, tanımından da anlaşıldığı üzere, en iyi sonuca, en iyi hedefe ulaşmaktır. En iyi sonuca ulaşmak için de mevcut durum ya da durumlar üzerinde iyileştirmeler gerçekleştirilebilir. Bu noktada mühendislik devreye girer

Optimizasyonun Aşamaları

Bir sistemin tasarlanma sürecinde önce sorun gözlemlenerek ihtiyaçlar belirlenir. Daha sonra bu ihtiyaçlar doğrultusunda sistem tasarımı yapılarak denemelerle iyileştirilir. İhtiyaçlar belirlendikten sonra ise sistemin çalışması dahil olmak üzere tüm aşamalarda optimizasyon devreye girer. Bir sistem var oldukça, optimizasyon da o sistem için var olacaktır.

* Sistemin temelinin oluşturulması
* Tasarım değişkenlerinin tanımlanması
* Hedef fonksiyonun belirlenmesi
* Kısıtların belirlenmesi
* Uygun optimizasyon metodunun seçilmesi ve uygulanması

**Optimizasyonun Kullanım Alanları**

* Yapı-Araç İskeleti Dinamiği’ne ilişkin problemler
* Dizayn problemleri
* Ekonomi problemleri
* Zaman ve maliyet problemleri
* Operasyon araştırmaları
* Benzetim (Simülasyon) ile Senaryo Analizleri

En basit anlamı ile **optimizasyon** eldeki kısıtlı kaynakları en optimum biçimde kullanmak olarak tanımlanabilir. Matematiksel olarak ifade etmek gerekirse optimizasyon kısaca bir fonksiyonun minimize veya maksimize edilmesi olarak tanımlanabilir. Diğer bir değişle optimizasyon **“en iyi amaç kriterinin en iyi değerini veren kısıtlardaki değişkenlerin değerini bulmaktır”**. Başka bir tanımlama ile **“belirli amaçları gerçekleştirmek için en iyi kararları verme sanatı”** veya **“belirli koşullar altında herhangi bir şeyi en iyi yapma”** olarak da tanımlanan optimizasyon kısaca **“en iyi sonuçları içeren işlemler topluluğudur”**.

Optimizasyonda bir amaç da maksimum kar veya minimum maliyeti sağlayacak üretim miktarını kısıtlara bağlı olarak tespit etmektir. Günümüzün bilgisayar teknolojisi kadar güncel bir kavram olan optimizasyon kavramı çok çeşitli endüstri kesimlerinde uygulama olanağı bulmuştur.

Optimizasyon, en genel anlamıyla, bir sistemde, belirli kısıtlar altında, belirlenmiş bir amaç fonksiyonunun değerinin en iyilenmesi amacıyla karar değişkenlerinin alacağı değerleri belirleme işlemidir. Diğer bir ifade ile, istenen bir çıktıyı elde etmek amacıyla, sistem girdilerinin ve/veya bu girdilerin değerlerinin ne olacağını belirleme sürecidir. Amaç fonksiyonunun en iyilenmesi demek, problemin türüne göre en küçüklenmesi (minimizasyon) veya en büyüklenmesi (maksimizasyon) olabilir. Örneğin bir çizelgeleme probleminin, geciken iş sayısını en küçüklemek amacıyla çözülmesi bir optimizasyon problemidir. Aynı şekilde, bir montaj hattı dengeleme probleminin istasyon sayısını minimize etmek (veya hat etkinliğini maksimize etmek) amacıyla, veya bir ekonomik stok miktarı belirleme probleminin toplam maliyeti minimize etmek amacıyla çözülmesi birer optimizasyon işlemidir.



Simülasyon, girdilerin bilindiği bir sistemde, çıktının tahmin edilmesi, belirlenmesi, sürecidir. Örneğin, siparişlerin geliş zamanının ve sistemde geçireceği sürelerin bilindiği bir üretim sürecinin simülasyonu sonucu, üretimin ne zaman tamamlanacağı, sistemdeki boş zamanlar, sistemin etkinliği vs. büyük oranda hesaplanabilir (tahmin edilebilir).

Optimizasyon ise, istenen çıktıyı elde edebilmek amacıyla, girdilerin veya bu girdilerin değerlerinin ne olacağının belirlenmesi sürecidir. Örneğin, istenen yüzey pürüzlülüğünün elde edilebilmesi amacıyla, bir 3B yazıcının parametrelerinin ne olacağının belirlenmesi optimizasyona örnektir.

Karar değişkeni, problem çözüldüğü zaman değeri belirlenecek olan sistem öğesidir. Yani aslında karar değişkenlerinin alacağı değerlerin belirlenme süreci, problem çözme sürecidir. Karar değişkenlerinin alacağı değerler, amaç fonksiyonunun değerinin ne olacağına etki eder. Fakat bu süreçte bazı kısıtların sağlanması gerekmektedir.

Çözüm uzayındaki, problemin kısıtlarını sağlayan herhangi bir çözüm, **uygun çözüm** olarak adlandırılır. **Optimum çözüm** veya **optimal çözüm** ise, uygun çözümler arasından, belirlenen amaca göre en iyi çözümü ifade etmektedir. Yani uygun çözüm, doğrusal programlama probleminin tüm kısıtlarını doyuran çözüm iken optimal çözüm tüm uygun çözümler arasında amaç fonksiyonunu iyi karşılayanı çözümdür.

### **1.2. Doğrusal Programlama**

Bir doğrusal programlama problemi genel itibari ile amaç fonksiyonu ve doğrusal sınır/sınırların yer aldığı iki kısımlı bir matematiksel ifadedir. Bu matematiksel ifade ile bir amaç maksimize ya da minimize edilir. Doğrusallık ifadesi modelde yer alan tüm değişkenler (fonksiyonlar) arasındaki ilişkinin doğrusal olmasından kaynaklanmaktadır. Doğrusal sınırların oluşturduğu kesişim kümesinden yola çıkılarak mümkün çözümler ya da uygun çözüm alanı belirlenir. Belirlenen uygun çözüm alanı ise amaç doğrultusunda eniyilemeye çalışılır.

Doğrusal programlama (DP) ile bağımsız değişkenlerden oluşan bir dizi fonksiyon ile yine bir dizi bağımsız değişkenlerin bir fonksiyonu olan bağımlı değişkenin optimal değeri belirlenmeye çalışılır. Bir başka ifade ile doğrusal programlama, belirli bir amacı eniyilemek maksadıyla sınırlı kaynakların nasıl dağıtılması gerektiğine çözüm arayan bir karar verme aracıdır.

Doğrusal programlama, karşılıklı ilişkileri doğrusal nitelikte olan ve sınırları kısıtlı kaynaklarla çevrili iki veya daha fazla değişkenlerin, belirlenen amacı eniyileyecek şekilde miktarlarının bulunması yöntemidir. Burada; eniyilenmek (max/min) üzere tek bir amaç, bu amacı sınırlandıran kısıtlar ve amaca ulaşmak için alternatif yollar mevcuttur. Bu üç öğe birlikte doğrusal programlama tekniğinin ana çerçevesini oluşturmaktadır.

Bir Doğrusal Programlama Modeli, doğrusal kısıtlar altında doğrusal bir fonksiyonun değerini maksimize ya da minimize etmeye çalışır. Doğrusal Programlama belli bir amacı gerçekleştirmek için sınırlı kaynakların etkin kullanımını ve çeşitli seçenekler arasında en uygun dağılımını sağlayan matematiksel bir tekniktir.

#### **1.2.1. Bir Doğrusal Programlama Modelinin Genel Yapısı**

Amaç Fonksiyonu:

Kısıtlar:

Karar Değişkenleri:

Pozitif Koşulu:

Bu formülde kullanılan matematik notasyonların açıklamaları ise şöyledir:

• (kısıt sayısı)

• (karar değişken sayısı)

• = Karar değişkenleri

• = Amaç fonksiyonu katsayıları, j karar değişkeninin katkı katsayıları

• = i kısıtındaki j karar değişkeninin teknik katsayısı

•= i kısıtının sağ taraf değeri (STD) yahut kaynak değeri

Kapalı matematiksel formülü verilen bir doğrusal programlama modelinin cebirsel açılımı ise şöyledir:

![metin içeren bir resim

Açıklama otomatik olarak oluşturuldu](data:image/jpeg;base64,/9j/4AAQSkZJRgABAQEAYABgAAD/4RDgRXhpZgAATU0AKgAAAAgABAE7AAIAAAAHAAAISodpAAQAAAABAAAIUpydAAEAAAAOAAAQyuocAAcAAAgMAAAAPgAAAAAc6gAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGxlbm92bwAAAAWQAwACAAAAFAAAEKCQBAACAAAAFAAAELSSkQACAAAAAzAxAACSkgACAAAAAzAxAADqHAAHAAAIDAAACJQAAAAAHOoAAAAIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyMDIwOjA0OjI4IDE1OjQ1OjQzADIwMjA6MDQ6MjggMTU6NDU6NDMAAABsAGUAbgBvAHYAbwAAAP/hCxlodHRwOi8vbnMuYWRvYmUuY29tL3hhcC8xLjAvADw/eHBhY2tldCBiZWdpbj0n77u/JyBpZD0nVzVNME1wQ2VoaUh6cmVTek5UY3prYzlkJz8+DQo8eDp4bXBtZXRhIHhtbG5zOng9ImFkb2JlOm5zOm1ldGEvIj48cmRmOlJERiB4bWxuczpyZGY9Imh0dHA6Ly93d3cudzMub3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMiPjxyZGY6RGVzY3JpcHRpb24gcmRmOmFib3V0PSJ1dWlkOmZhZjViZGQ1LWJhM2QtMTFkYS1hZDMxLWQzM2Q3NTE4MmYxYiIgeG1sbnM6ZGM9Imh0dHA6Ly9wdXJsLm9yZy9kYy9lbGVtZW50cy8xLjEvIi8+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczp4bXA9Imh0dHA6Ly9ucy5hZG9iZS5jb20veGFwLzEuMC8iPjx4bXA6Q3JlYXRlRGF0ZT4yMDIwLTA0LTI4VDE1OjQ1OjQzLjAwOTwveG1wOkNyZWF0ZURhdGU+PC9yZGY6RGVzY3JpcHRpb24+PHJkZjpEZXNjcmlwdGlvbiByZGY6YWJvdXQ9InV1aWQ6ZmFmNWJkZDUtYmEzZC0xMWRhLWFkMzEtZDMzZDc1MTgyZjFiIiB4bWxuczpkYz0iaHR0cDovL3B1cmwub3JnL2RjL2VsZW1lbnRzLzEuMS8iPjxkYzpjcmVhdG9yPjxyZGY6U2VxIHhtbG5zOnJkZj0iaHR0cDovL3d3dy53My5vcmcvMTk5OS8wMi8yMi1yZGYtc3ludGF4LW5zIyI+PHJkZjpsaT5sZW5vdm88L3JkZjpsaT48L3JkZjpTZXE+DQoJCQk8L2RjOmNyZWF0b3I+PC9yZGY6RGVzY3JpcHRpb24+PC9yZGY6UkRGPjwveDp4bXBtZXRhPg0KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICA8P3hwYWNrZXQgZW5kPSd3Jz8+/9sAQwAHBQUGBQQHBgUGCAcHCAoRCwoJCQoVDxAMERgVGhkYFRgXGx4nIRsdJR0XGCIuIiUoKSssKxogLzMvKjInKisq/9sAQwEHCAgKCQoUCwsUKhwYHCoqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioqKioq/8AAEQgBHQOEAwEiAAIRAQMRAf/EAB8AAAEFAQEBAQEBAAAAAAAAAAABAgMEBQYHCAkKC//EALUQAAIBAwMCBAMFBQQEAAABfQECAwAEEQUSITFBBhNRYQcicRQygZGhCCNCscEVUtHwJDNicoIJChYXGBkaJSYnKCkqNDU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6g4SFhoeIiYqSk5SVlpeYmZqio6Slpqeoqaqys7S1tre4ubrCw8TFxsfIycrS09TV1tfY2drh4uPk5ebn6Onq8fLz9PX29/j5+v/EAB8BAAMBAQEBAQEBAQEAAAAAAAABAgMEBQYHCAkKC//EALURAAIBAgQEAwQHBQQEAAECdwABAgMRBAUhMQYSQVEHYXETIjKBCBRCkaGxwQkjM1LwFWJy0QoWJDThJfEXGBkaJicoKSo1Njc4OTpDREVGR0hJSlNUVVZXWFlaY2RlZmdoaWpzdHV2d3h5eoKDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uLj5OXm5+jp6vLz9PX29/j5+v/aAAwDAQACEQMRAD8A9W8Qy+Kf7ST/AIR2XR0tRGNwvoZWcvk5wUYDGMfrWcbn4iFcfaPDA+ltcf8AxyunuV2uv+7moa5aNKDgm0aSk1LQ51Lj4hoMCfwx+Ntcf/HKTz/iHv3G48Mk/wDXtcf/AByujorX2NPsTzy7nOtc/ERhg3Hhn8La4/8AjlKLr4iKMCfwx/4DXH/xyuhoo9jT7Bzy7nOrcfENST9o8Mkn1trj/wCOUPcfEN8ZuPDPH/Ttcf8Axyuioo9jT7Bzy7nPfaviJjH2jwx/4DXH/wAcpFufiGgwLjwx/wCA1x/8croqKPY0+wc8u5zpuPiGW3G48M/+A1x/8cpTdfERgR9o8Mf+A1x/8croaKPY0+wc8u5zwuviIvSfwx/4DXH/AMcpBc/EQMT9o8MZ/wCva4/+OV0VFHsafYOaRzzXXxEbGbjwx/4DXH/xyl+1/ET/AJ7+GP8AwGuP/jldBRR7Gn2Dmkc8t18RF6XHhj/wGuP/AI5Qbr4iEg/aPDHH/Ttcf/HK6Gij2NPsHNI543fxEP8Ay8eGP/Aa4/8AjlAu/iIBjz/DH/gNcf8Axyuhoo9jT7BzSOe+1fETdn7R4Y/8Brj/AOOUG7+IhH/Hx4Y/8Brj/wCOV0NFHsafYOeXc5/7X8RP+e/hj/wGuP8A45SC6+IgOftHhj/wGuP/AI5XQ0Uexp9g5pHPG6+Ih/5ePDH/AIDXH/xyl+1/ET/nv4Y/8Brj/wCOV0FFHsafYOaRzwu/iIP+Xjwx/wCA1x/8co+1/ETOfP8ADH/gNcf/AByuhoo9jT7Bzy7nP/a/iIf+Xjwx/wCA1x/8co+1/ET/AJ7+GP8AwGuP/jldBRR7Gn2Dnl3Of+1/ETP+v8Mf+A1x/wDHKDd/EQ/8vHhj/wABrj/45XQUUexp9g5pHP8A2v4if89/DH/gNcf/AByj7X8RP+e/hj/wGuP/AI5XQUUexp9g5pHP/a/iJ/z38Mf+A1x/8co+1/ET/nv4Y/8AAa4/+OV0FFHsafYOaRz/ANr+In/Pfwx/4DXH/wAco+1/ET/nv4Y/8Brj/wCOV0FFHsafYOaRz/2v4if89/DH/gNcf/HKPtnxE/57+GP/AAGuP/jldBRR7Gn2Dmkc/wDbPiJ/z38Mf+A1x/8AHKPtnxE/57+GP/Aa4/8AjldBRR7Gn2Dmkc/9s+In/Pfwx/4DXH/xyj7Z8RP+e/hj/wABrj/45XQUUexp9g5pHP8A2z4if89/DH/gNcf/AByj7Z8RP+e/hj/wGuP/AI5XQUUexp9g5pHP/bPiJ/z38Mf+A1x/8co+2fET/nv4Y/8AAa4/+OV0FFHsafYOaRz/ANs+In/Pfwx/4DXH/wAco+2fET/nv4Y/8Brj/wCOV0FFHsafYOaRz/2z4if89/DH/gNcf/HKPtnxE/57+GP/AAGuP/jldBRR7Gn2Dmkc/wDbPiJ/z38Mf+A1x/8AHKPtnxE/57+GP/Aa4/8AjldBRR7Gn2Dmkc/9s+In/Pfwx/4DXH/xyj7X8RP+e/hj/wABrj/45XQUUexp9g5pHP8A2z4if89/DH/gNcf/AByj7Z8RP+e/hj/wGuP/AI5XQUUexp9g5pHP/a/iJ/z38Mf+A1x/8co+2fET/nv4Y/8AAa4/+OV0FFHsafYOaRz/ANr+In/Pfwx/4DXH/wAco+1/ET/n48Mf+A1x/wDHK6Cij2NPsHNI5/7X8RP+e/hj/wABrj/45R9r+In/AD38Mf8AgNcf/HK6Cij2NPsHNI5/7X8RP+e/hj/wGuP/AI5Qbv4iH/lv4Y/8Brj/AOOV0FFHsafYOaRz/wBr+In/AD38Mf8AgNcf/HKBd/EQf8t/DH/gNcf/AByugoo9jT7BzSOe+1/EQnPn+GP/AAGuP/jlL9r+In/Pfwx/4DXH/wAcroKKPY0+wc0jnxd/EQD/AF/hj/wGuP8A45Sfa/iJnP2jwx/4DXH/AMcroaKPY0+wc0jnzd/EQj/j48Mf+A1x/wDHKPtfxE/57+GP/Aa4/wDjldBRR7Gn2Dmkc8Lr4iA/8fHhj/wGuP8A45Qbr4iHH+keGP8AwGuP/jldDRR7Gn2Dmkc/9r+In/Pfwx/4DXH/AMcpBdfERelx4Y/8Brj/AOOV0NFHsafYOaRz32r4ibs/aPDH/gNcf/HKDdfEQjBuPDH/AIDXH/xyuhoo9jT7BzSOeF18RAMfaPDH/gNcf/HKBdfEQMT9o8Mc/wDTtcf/AByuhoo9jT7BzSOea6+IjDBuPDH/AIDXH/xyl+1/ETH+v8Mf+A1x/wDHK6Cij2NPsHPLuc6tz8Q1zi48Mf8AgNcf/HKGuPiGxBNx4Y4/6drj/wCOV0VFHsafYOeXc543XxEK4+0eGP8AwGuP/jlItz8REXAuPDH421x/8croqKPY0+wc8u5zv2j4h7932jwxn/r2uP8A45Q1z8RHXBuPDH/gNcf/AByuioo9jT7Bzy7nPC6+IgAH2jwx/wCA1x/8cpqz/ENWLC48M5P/AE7XH/xyujoo9jT7Bzy7nOPN8Q5MZuPDI+ltcf8Axyjz/iGV2/aPDOP+va4/+OV0dFL2NPsPnl3OZR/iEnS58M/+A1x/8XQz/EJm3G58NZ9ra4/+LrpqKPYUuwc8u5zbS/EJlwbjwzj/AK9rj/45SI/xCQYFz4Z/G2uP/i66Wij2FPsHPLucxn4g7932nw1n/r2uP/i6V2+IMgwbjwz+Ftcf/F101FHsKX8oc8u5zQk+IQXH2jwz/wCA1x/8cpqf8LBQki58Nf8AgNcf/F109FHsKX8oc8u5zDn4gu2TceGh9La4/wDi6cX+IRXabjwzj/r2uP8A4uuloo9hS7Bzy7nMIfiDH0ufDX421x/8XSH/AIWAW3fafDWf+va4/wDi66iij2FL+UPaS7nMs3xBdcG48M/+A1x/8XQrfEFRxc+Gv/Aa4/8Ai66aij2FLsHPLucuB8QA2Rc+Gv8AwGuP/i6V/wDhYD4zc+GuPS2uP/i66eil9Xpfyh7SXc5kN8QQuPtPhnH/AF7XH/xdEA8eJMhluPDhj3DeFt58474+frXTUU/YUv5Q55dyy8YZs0U5SCoorw7I7Rt//rx/u/1NVqmum3zZ9qhr3KP8NHFP4mFeYeM/GWueEfiFpsj3Qfw1NLHBdxtEmYmkDYO4DOOM9f4T616fXH+KvB83ivT9d0+8W2jgvoIxayiRmdJUyVZl2jAyR0J4zWuqaaJVnozD8SeMNbh+Kvh/R9Nu1g0m9ne3l2xIzSMgBYhmBwMsF47qa6g+PNBGrxaeLpmeW8awWUJmP7QoBMZPY8jtj3rntR+H2pnxL4Pu9NltHtPD6sJjcTuJZywG5uEIyTk8nnNZt78PvFF94lt9UuH01/suvC+jVJmRTbjtsCYD8ctyW7ngVUbaJ93+a/S7Jffy/R/8A7S28caRd63c6RB9pa/tZlhlgMBDKzAkHnquFJ3DjHeqGvaprM514aHffZTYpBbW5MSMpuHILE7gcgK8Yx9apX/hHxC/xJs/FmmHTYHQNbXUDTvie2wMciP7+cn04UZ4rYh0PUofCr2yrbPqNzdm6uN8rKm4y7yAwUk4ACjjtU2vHX+v6RWz0/r+vyOW8NfEHVdX+GV5NOyL4ls5pLOUGMDbIAW3lenCAnHTKmq/h7xrql98P/DGq6vr7Wl3qep+S7iwSRZgJGUQgADZuA+92wa3x8PRaeJPEmt2DRebrFoI0t3crHHMVIdyQD1yOcZ5b1rn7T4ZeIbfwL4Y0RpdMM+i6qL6WQXEm2RA7Nhf3ec/N39KpO7u/wC7+ev+ZMtrLz/LT/I6q4+J3hu0vLm2uJrlHtLxbO4b7M5SGRjgbmxjBx1rpr+/h02za5uEuHjUgEW1tJO/P+xGrMfyry3WPhf4h1CLxGkM2mL/AGtrEN/CXuJPkRCchv3fXkdMivUL7TrbVbE2uowiSJsFkDkcj3GDSXw36/8AAX63G/i8v+C/+AUbPxRYX13HbQW+qrJIcAz6PdwoPq7xhR+JrZrFsvCGh6deR3VnY+XNGco3nOcfgWxW1QAUUUUAFFFFABWD/wAJlozazFpsc7SSy3DWqyIhMfnKpYpu7kAc4yB0JBrckUvEyq21mUgN6e9ch8PdA1rw1oMOkavFYMlrJKy3UMzO85Zy24qVG0/NzyaFuD2N7xBr9l4Z0eXVNUMq2sJHmPFEX25OMkDtz1pdN16z1a8uYLETuLcKTOYWWKTdn7jkYfGOcdK4y1h1/wAaafrVtLcW76FqU9zbxzzHMsUaHywEQKAQSrHcTnkdcVNu8QeEfhEbXUJrZNVtYUsrOSzO/c5YRxk71xk5XjHbr6C2v6fiN729Tv6KhtY5IbOGKaVppEjVXkbq5A5J+tTU3uStUFFFFIYUUUUAFMmmjt4XmnkSKKNSzu7BVUDqST0FPpGVXUq4DKRggjINAHM2njrRbnxDeWB1fTBDDFC0Mou0/es5fKjnBxtHT1rp6x7bw5aW+v3upeXCwuooY1i8kDyym/nPvv8AQdK2KA6hRRRQAUUUUAFFFFAFPVNVstGsHvNSnEMCkLkgksxOAoA5JJ4AHJqHRtds9diuXsvMBtZ2t5kkXBSQAEj07j+XWsXxx4f1TWptCutHNvK2l6gt3Ja3MhjSYAEfeAOCM5HFL4rm1GPw4sdvqcelavMxMC27rsL5/iZ15QdWOAT+IBFtr/W3/BDqv67/APALN5440ax1S/06c3RvbGJZWt47Z3eVW6eWoGX6dulb8MgmhSVQyh1DAOpUjI7g9D7VwLeGPFUvjLRvFMjaWl9HbvZ31urv5YiPIZW25Zg2TjgdB6mttJruf4gBbPWGks4bdxd2bFNqv8uwIAN2RyWJJ6gfQXbrr/X5A+501FFFABRRRQAVFc3VvZWz3F5PHbwxjLyyuFVR7k8Cpaa8aSoUkRXU9VYZBoA53SfG2kanrF/YrqmnZhuUhttl0ha43Ro3Azz8zEceldJWRpvh610/VNRvFSFzeTpMiiEDydsapgHv93PbrWvR0QBRRRQAUVzd9J40Dan/AGdDohAlj/s/znl+aP8A5aeZjv6YpLyTxqP7S+wQ6GcSxfYPOeX5k/5aeZjv6YoA6WiuavJPGo/tL7BDoZxLF9g855fmT/lp5mO/pitLTm1w6pqA1VLFbEMn2E27OZCMfN5meOvTFAGnRRRQAUjMEUs5CqoySTgAUtIRkYPIoA5gePND/wCEkNh/bGl/Zvsgm8/7YmN+/G3OcdOa6dWDKGUggjIIPWsf/hG7T/hIzqnlwbTaC38jyBjO/duz+mMVsAYGBwKOi/rqHX+uwtFFFABRRRQAUUUUAFFFFABRRRQBBeXltp1lNeX0yQW8CF5JZDhVUdSaz9G8S6frl1c29kZRLarG8iSxlCFkBKHB6ZAzg4PqOazviF4dv/FHhGXTtKlhS486KXZOSElCOGKMQCQDj0pus+KLzw5Y2UuoabbNdXtx5X2e3uCRGixs7OXKDcFCk9Bj1pXSTbHbsaOreKtK0PVbKw1OV4Jb4OYXMZ8s7F3EbumcdquaVqcWr6el5bxXMMbswCXMDQuMEjlWAI6VwGqaZ4m+IPhvTNYjt9P024tbiDUdNgeVnL9DiVto28E8AHPGcVsav47udG1yHSX0tLu4K2wleG4Kqkk0pQIMqc4AZs8ZA6Cqs9nvexLfVbf1/wAA7SiiikMKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACiiigAooooAKKKKACorh5I7WV4Y/MkVCUTONxA4FS1BepNLp9xHavsnaJhGxONrEcH86UtmNbnnHgfx6/iprS3Otm31uOYjUNJuoEj+UZ3CL5QeOO5OAc+tO1zWPEmi+OvCWhPrski6t5v2x1t4RgqMjZ8nA575q9feB7nXNd0HVdSstPtNT0y5jnudQtZCWuQg+5jaDgnH3jwOBnNT+KfCWraz8Q/Deu2RshaaOZDKk0zrJJuAHACEce5ql8Ub99fQn7L9DM8Va14j8P+JvB+kLrTONVvJYLqUW8WWQONhGUwG2sAe2R0rQ8PeLtQX4lar4M1t0uXt4RdWd2qBGkjOMq4HG4buoAzg8U7xt4S1jX/F3hjVdMNiIdFuGmkS4ndGkyV4GEYfw1Y0LwZPb+PNT8XazNC1/eRrbwW9uSyW8QA/iIBZjtHOB3pQ218/+B+I5eXl+ev4HYUUUUASRv8goqI5RiPeivm+Y9Eo6/da/aXsaaN4eGp2/lBmmN8kO1snK7WGTxg596yk1Xxk/3fBI/wDBtF/hXbztwEHenoBHHz6V3xxE4rlXQ53BN3OEbVvGKsFPgpc/9haL/CnHU/GYGT4JXH/YXi/wrtIhvkLnp2p07Zwi96r61UsL2cTiF1Xxk4yvgkY/7C0X+FB1XxkG2/8ACErn/sLRf4V3YxHH9BUcI3MXP4UfWqnkHs4nEnU/GajJ8FLj/sLxf4Ui6p4zYZHglcf9haL/AArt5TuYIPxqThE9gKf1qoHs4nCf2r4y3bf+EKXP/YWi/wAKVtT8ZqMnwSuP+wvF/hXbwjOWPeiQ7nCj8aX1qY/ZROIGp+MyMjwSv/g3i/wpP7V8ZFtv/CErn/sLRf4V3hO1M+lMiHG49TT+tTF7OJwzap4yUZPglf8Awbxf4Uo1LxmR/wAiSv8A4N4v8K7dzukC+nWpGbauaX1qoHs4nBjVPGRYgeCV4/6i0X+FKdU8Zjr4JX/wbRf4V3UQ+XPc00/NKB2FP61UD2aOH/tLxn/0JK/+DeL/AAoGqeMz08Er/wCDeL/Cu7c4WiMYWj61MXs0cIdT8Zr18Er/AODeL/CkbUPGToVfwOjKwwQdWhII/Ku6+9J9KcxwtL61Mfs0ee2U3iiwtVttO8AW9rbpnbFBqcCKvc4AXFJeSeJr9Y0v/h9bXIjcSRibUoH2sOjDK8EetehoMKKQDMh9qf1qoL2aOF/tLxn/ANCSv/g3i/woGp+Mz08Er/4N4v8ACu7c4X9KVRhQKPrUw9mjg/7T8Zg4/wCEJX/wbxf4Uv8AaXjP/oSV/wDBvF/hXdDmQ+1Dnj60fWph7NHCDUvGZ6eCV/8ABvF/hR/afjPP/Ikr/wCDeL/Cu8AwKaOWJo+tTD2aOF/tLxn/ANCSv/g3i/woGpeMz/zJK/8Ag3i/wru26Uo4FP6zMfs4nB/2l4zz/wAiSv8A4Nov8KP7T8Zj/mSV/wDBvF/hXdryTQfSl9amHs4nC/2l4z/6Elf/AAbxf4Un9peM8/8AIkr/AODeL/Cu9pB6+tH1qYvZo4P+0vGY/wCZJX/wbxf4Uv8AaXjT/oSV/wDBvF/hXdnsKWj61MPZo4L+0vGf/Qkr/wCDeL/Cj+0vGf8A0JK/+DeL/Cu8FB6iq+szD2aOE/tLxp/0JK/+DeL/AAqheWuvajewXl/8NbG6urcgwzz39u7xYORtYrkc88V6XSCl9Zmg9mjhP7S8af8AQkr/AODeL/CqFja69pl1Pc6b8NbG0nuDmaW3v7eNpec/MQuTzzzXpdFH1mYezRwf9peNP+hJX/wbxf4Uf2l40/6Elf8Awbxf4V3lFCxMw9mjg/7S8af9CSv/AIN4v8KP7S8af9CSv/g3i/wrvKKf1mYezRwf9peNP+hJX/wbxf4Uf2l40/6Elf8Awbxf4V3lFL6zMPZo4P8AtLxp/wBCSv8A4N4v8KP7S8af9CSv/g3i/wAK7yil9amHs0cH/aXjT/oSV/8ABvF/hR/aXjT/AKElf/BvF/hXeUUfWph7NHB/2l40/wChJX/wbxf4Uf2l40/6Elf/AAbxf4V3lBprEzD2aOD/ALS8af8AQkr/AODeL/Cj+0vGn/Qkr/4N4v8ACu8oo+tTD2aOD/tLxp/0JK/+DeL/AAo/tLxp/wBCSv8A4N4v8K7yij6zMPZo4L+0vGf/AEJK/wDg3i/wpf7S8af9CSv/AIN4v8K7sdaCaPrMw9mjhP7S8af9CSv/AIN4v8KT+0vGf/Qkr/4N4v8ACu9pAOtH1mYezRwf9peMx/zJK/8Ag3i/wpf7S8af9CSv/g3i/wAK7s0val9amHs0cENS8Zn/AJklf/BvF/hR/aXjMf8AMkr/AODeL/Cu7Xv9aG7UfWph7NHC/wBpeNP+hJX/AMG8X+FINT8Zn/mSV/8ABvF/hXe01e/1o+tTD2aOE/tPxmP+ZJX/AMG8X+FL/aXjP/oSV/8ABvF/hXdH7w+tOp/WZj9nE4Ial4zI48Er/wCDeL/Cg6n4zBx/whK/+DeL/Cu8T7tNf74pfWpi9mjhv7S8Z/8AQkr/AODeL/Cqt03ibUNpvfh9bXG1XRTLqUDFVYYYDK8Ajg+or0X+GkT7tP6zMfs0cBDeeLrWGO3t/AscUUahUjTVYQqgdAABxVeaHxDPcG4n+HNnJMZElMjajAWLoCFbO3OQCQD2zXoxHzrTmHBpfWqm4vZxODGpeMyOPBK/+DeL/Cj+1PGe7H/CErn/ALC0X+Fd2n3RTWOJR9KPrUx+zicMdT8ZgZPglf8AwbRf4UDU/GZGR4JX/wAG8X+Fd3J/qzSRNlBR9ancPZo4T+1PGW7b/wAISuf+wtF/hQ2qeMlGT4JX/wAG0X+FduTi4p8w/dUvrVSwezicMNT8ZsMjwSv/AIN4v8KaNW8Ylio8FLkf9RaL/Cu7hOYxUY4uD70fWqgeyicS+q+MkGW8FL/4Nov8KUan4zIyPBK4/wCwvF/hXb3I/d0sJzGM0/rVS9g9nE4VdX8Ys2B4KGf+wtF/hQ+reMo/veCR/wCDaL/Cu1Ti4NPuhmMGp+t1LdA9lE4j+1PGRXP/AAhK4/7C8X+FIur+MXOF8FD/AMG0X+Fd1Cd0WKgj+Wcij63U8h+yicY+r+MUOG8FD/wbRf4Uv9q+Mtuf+EJXH/YWi/wrtbocA+9Pj+aL8KPrdS9tA9lE4VNY8Yufl8FD/wAG0X+FDax4wRsN4KGf+wtF/hXZwnbKwpbocqwpfW6lr6B7KNzjDq3jILn/AIQkY/7C0X+FImseMH+74KH/AINov8K7hPmh/CoIDtkIp/W6l+geyica2s+MFbafBQz/ANhaL/CnHV/GKrk+Chj/ALC0X+FdjcjDKwqQfNDj2o+t1PIPZROJTWfGD/d8FD/wbRf4Uh1rxeG2nwUM/wDYWi/wrsbc4Yii5G2QEVP1ypa+g/YxOPbWPGKrk+Chj/sLRf4Ui6z4wcfL4KH/AINov8K7UjfDUVvw2Pen9cqeQeyicd/bfi8Pt/4QoZ/7C0X+FK2s+MEXLeChj/sLRf4V19wNsoNSMN8H4UvrlTyD2MTjF1nxg4+XwUP/AAbRf4Un9ueL923/AIQoZ/7C0X+FdhbNyRTZhsmB9aPrlS19A9jE5J9a8YIMt4KGP+wtF/hQus+MGGR4KH/g2i/wrspBvgqK3cbtmRnHTNH1ypfoHsonIDXPFxbH/CFDP/YWi/wpX1vxemN3gof+DaL/AArrZBsnz2NSSjfBmj65U8g9lE48a14wYZHgoY/7C0X+FLBrfiuWdEk8HCNCwDP/AGpEdozycY5rrbdsjBqNhsn9jS+uVLdA9jEfLHlyaKnHIBorj5UbXJIh5khdvwp07cBB1NPQeXHz6VFEPMlLnp2rUglQCOP8KjiG+QufwpZ36KvU09QEjGfxp9QGTtkhB3PNSAbI/TAqKL55C5/CnTtkBB1NHmAkI3MXP4Usp3MEHenqNkf0FRxAtIWNFtLASjCp9KjhG5i5omboo71Io2rTAjlbLBR3qT7qfSo4xucufwpZTnCDqaXmIIhnLnvRIdzKg+pp4G1QKZGNzFvyoAf91fpSRjgk96bKc4UdTUg4WmBG53OFp5OFpkfzMWpZD0WgCnqmq2+h6Pcale7vKhXO1BlnJOFVR3JJAA9TWVpniDUL7xTPo97p8MBtrOO6meO4MnltIxCxn5RzhWJPTpV/xFoMPiHRjYTXE1rtlinjmg27keNw6n5gQeVHBBrMs/Dr+H213VtMkvNS1G/QOIpnjG+RYwq84UDJHc4GTjAqk1yu+/8AX9fJCd27I2l1rTnt7eaO7jaO5lMMLA8SOCQVHv8AKfyNYejeJdX1LXLvTG0e3hk0+6Ed7J9sLKsTIHjeP5AXJzyCFAweTUB8PXuzQdLi8+2t9PspGa9QxlkuSgjUgHOTh5WyQRnFaen+FrfSdam1a3uryWee3WO4jLLi4dSxEjcD5vnI6henHAxdopt7/wBf196Fdtaf1/X6Grb3lveySrayrL9nkMUu3+BwASp9+R+dWScDNYvhOyubPQlbUITDeXMslxcRlgxV3csVypIOMgcHtWw56D1rN6DFQfLz3oPLgelL0FNTuaAHHgUijC0N2HrTqBjerfSklfy4XfazbQTtUZJ9hSr3NDdhQBztr4hmk8R3dq1jf+QltA6J9nGUZmlDE98EKv5V0X8VRLawx3st2qYmljSN3yeVUsVGOnBdvzqUdc03YQp6UDgCkbsKWkAZ+ag9KB60hPagBRQOtLSCgDl/EHjH+yV1U2Volyuj232i9eSXYFJG5Yl4OXIGewGV9a2rbUV8mwTUNlteXkYK2+7PzhNzKD3xz+Vc9qnw9s9Um1TzNU1CKDU7mG7lt0MexZY9gyMoSQRGowSR7VozWt3/AMJVHdmzmmtdO091t28xCZ5XYbgMsMEKgGWwPnPNX7tv67f56Cd7kes+LYrbQdT1DQPsurSaWC11AtzsKKBubGFbLYBwOAfWpbfxKkWlyarr32XTNOkZWtJXuMmSNlyGcYAU9TgE4HeqVnoUutTzahrEN7p5uP3dzp8jQsk6KBsDFC/yrlujDJLZGDiiTSrnRPDdn4dsLTUtWtWi+zy3RlgDRRYC4O5k/h4yoPQnk9Wkkrdf6uHU6pWV0DoQysMgg5BFLTIo0ghSKJQkcahVUDgAcAU+oe+g1e2oUUUUgCorib7PA0vlSS7f4I1yx+gqWimM57R9emvdU1KCa0vdkd4I4i0AAjXyo2wT9WJ59a6GoYLWG2luJIU2vcSebKck7m2hc+3CgcelTU3rawgooopAFFZ1zpt1P9t8vV7uD7SEEXlrGfs23qUypzu77s+2KLnTrqf7Z5Wr3dv9pCCPy1jP2bb1KZU53d92fbFGgGjRWdc6bdT/AG3ytXu4PtIQReWsZ+zbepTKnO7vuz7YqxBbyx31zO93NLHNt2QOF2Q4BB24GeepyT04xRoBYNLSE9KWmAgpGOBnGacOlIakZzjeI5x4lS2FjqHkG0ZzH9nGS28DPrjBNdEh3LnBGR0PaozaQm+F4U/fiMxB8n7pOcY6dQKlHSnpZL+txdX/AF0A9vrSmkPUUuaQxF6H60h6ihT1+tB6igOo6mr0/GnGmr0/GhbiBvvD60tNb7w+tOoGIn3fxpH++KE6fjSt95aQjJ1TW2tdTttKsYVnvriN5iJH2JDEuAXcgE9SABjk+gBIreHfE8eqeFLPWtRjTT0vJNsSGTeCGk2R84HLcEcfxCm614Og1nV5r86je2jXFg1hPHblAskeSR1UkEFj0Iz0ORUC6DJaW/hzRrOOafTtPlEk08jJwIkPlqRx/HtPAwNtaR5bK+//AA9/wsJ3vp/Wn+Z0N61zHbO1lFHNcBSY45ZDGrH0LAMR+RrE8J+I7/xPYxX76ZHZ2UkJBZrjdIJ1ba6bduNoIYbs5OOmOayLTQ/EV1q4fU9R1GygdZJXWK4RlO+ZiIcHdjagUblAPzHDcVck8K3GkWP2Tw7cXaQ3OpfaZFWVVFrHkyOqAY4Zhtwc/f8AQUKy3/r+twd76f1/X6nVxHK02T/XLWb4VhvLfw1ZR6oZTeCIGYSyb2DHkgtk5xnGcnpWlN/rFqGraFokbmM/SmQf6unnlT9Kig6Gl1F0CTidakk5jP0qObiRTUrcp+FJAR25/d0yTi4B9adb8Aikn4kU0ug+pJKMxGm25ylPbmM/Sorc8kU+odBj/LcippxmE1FccSqamb5ovwpd0HYZbH5Kib5bmnWp5IpLgYlBpdB9ThPCkiaN4Cl1Gx0c3d1NqF0srW0Y3kfaZQGcgFio9gx9BWzY6tdSabJq41O0uli+X7JbkrEhJAw77HkLDP8AdX3UUvgWC7svDkun39lPbSxXVzJukA2urzyMpUgnPBB/Gti3sLQ6h9t+zRfawuzzwgD7fQnqR7VpJrmJMGw8T3F1q0UD21miytglLiZiPoGgUfmRV3xDr02mTpDFBayAoHzLNKh6nskTjt6/hWrfKSGAZgWUgFCAw+me9VdBs7nT/DsFnf3k17PEGzPO4eRgWJUMwAyQCBnAzip91lajPDurSatbStLHDGUYKPJkd+3+3Gn8jWNd6tfeH9SFt9oXXGOCLWNAt4oPfCDYR7sIx6tWlqVndXs+nmzvbi0+zXSyy+VIFWVB95HGCWBGRgY55zxViO0tNOUR2FvFbxk7isahcnuT6n3pc0UrhZ7GLeeLrpWZPsdkuxiMPcz7h9cW5GfoSPc1UdbaX4heFtQS2t4rm6tbszSRLy/yR4BYqCwHbIH0rr3mDxY9q5O7ju7r4gaPNFaTi3sYrkS3DABMuqbQOcnoe1OM1zadn+TE07HWAhLg/Wn3LAxgg1TlYiQGnMS0fWsOboXYnhnGzBNRCTZMagiokBD55qeZtDsT3EpZQe9EdwTHioiNyUyMc0XdwsKJWSQ4NPmmdhk84qKQbWBp5wyUrvYZKl0xjxivNLXxBd6Z4p1nxDBpkM9ldanBpZl87ZIVVhHuRQpDYkd85I4Fehx+lYNl4TtdOW1iM89xb2kzTwQy7cLIxJ3kgAscscZ9c9eaunO3xEyTasjel1KCW7e2jkU3ESK7xjqqtnB/HB/Kp0uWKYI4rE0i0uP7Q1e9vYGhe4uFSEMykmFEAXoT1Yucdea1I+OKmTtsNEkc2yTmnTy7gGHaoJBiTNOxuSpu9h2RMl0Qo4oqoKKOdhY3bh+iL1NPRRHFUMQMsxc9O1PuHwoUdTXWu5kNiBklLn8KdO2FCjq1PjXy4xUSDzZS3YdKXQCWNQifhUceZJix7U6d8LtHU8U6JdkYp76ANnb5Qo6mnoNigVEv7ycnsKfO21MdzxR5gNT55i3YdKdO2FwOppYl2oKYPnmz2FHQCRF2oPpTE+eQt6U6RtqfWiJdqUxBM2EwOp4pyDagFRj55j6CnyNsTPegBq/PKT6UsjYX60sYwv603782Oy0hj1G1RTV+aQn0pznahpEGE+tMQSHC/WlUbVApp+aTHYU5jhSaAGj5pCfSnSHC0iLhfrSN8zgUdBDlGEApBy59qceFNNjHy5PegBWOFNKowoprDLgU48A0wGjmQ+1K5wtIg4+tDcsBQA5RgCk6v9KXoKavc0AObpQPu0jfeApaBiD730pTwKRe9DdhSAcOlIPvUUL3piA0AcCg0tACd6DSDvSt0oAB0oPWlpBTADQOlBoxRsAd6WkHWlo2AKKKTuaAFopDRTAWkNLSGkADpS0go/ipALSDpRQOlAAetLTf4qU0DAdKD1FA6UhzuFAC0L92g9KRfu0ABPzClpG/1gpe1ACJ92kb7wpU+7SN99aBD6ZGcj8aeaZGPl/GjqAPwV+tPqOTqv1p5oGNj7/WiT+H60R9/rSS/wAP1pdBDzTIv4vrT6ZF/F9aYwk/h+tP/hqOXoPrUnajqBFD1b60TDoaIv8AWNSz/dH1qeg+o/qPwqKDh2FSryoqGPiZqfVCFuOi/WpRyg+lR3A/d/jT4/8AVj6UdRkUBxIw96Lrop96RPluCKdc8x1PQOpIvMf4VBBxKwqaI5jFQr8twfemA66HAPvUicxU25GYqIDmOl9oOhDCdsxFOu/ug1Ex23JxS3DZj5NTfQZNG+Y6qq+2U0QS/LjNQuf39S5FJEty+VFMRiY+KWUZjpkXSpvqHQYhIkIp04yuaa/yzVI4zFSGJHytRqMSU+GmSDEmaQdR0w4BoQ5WnMN0dRxUdRjRxIcU+UZXNNk4kp7cx0AInK1H92TFOiPY02UYfNT0AfKMpmkjORg04jdHxUcZwcUdQE+7IadIMpTZRhgaeBlKAEjORTPuvRHw2KWUYYGkAsgymaSM5FPHKVEnDEUdQEdcMaKmZcnNFKwzZiASP9TUKDzZix5A6U+4fam0dTTol2RV29bGA2d9qbV6ninxIEjGagB82bPYVLPJsj46mhdx9Bi/vJ89hUkz7U46mkgXamT3qMt5txjstIfUmhTbH7moz+8n/wBlaklfZGSPoKbbrhMnvTt0EOkfbGaSFcJz1qOQ75gg6CpnbZHmgRGf3kwXstSO21CaZbjgse5pJjvkVB+NHS4x0K4XJ6mkc75AvYVIfkT6VFACSzHvQBKTtUk+lMiHBJ70kxyQg71IOFp9QGSHc4UfjUhOFqKIbnZ6dKeAPWjoBTv9TttH06W/vnKxIQAFGWdicKqjuSSAB6mqWmeIhq2sX2mGxubWewWIzeaUKgyAsFBVjk4AJ9MjrR4m0G51u10/7DeR2k9jfR3amaEyxvtyNrKGU98jnqBWLb6faeErPW7zxPrUckWq3hlMkj/ZmJKhVTfuHZQOCAAPrVK3K77/APDf8H7iXe6t/W/9fM7bOFrmrHxeLzWrnTIdG1D7XbXKw3CMI8QxsoKzE78bDngAluDxwayZYpY9O8N6FY6mJLmVnvEmaVpf3aKzLzkM6BniXk8qK1tM8NXuneJZNan1RJGurVI76MQbVkdCxVlyx2KA5GOTwOeua5Unr/X9bCu7af1/W50ch6D1NOHAFc74Ql+3219qiymWO+vpZIvn3KI1Plrj0BCbv+BGuhc4U1DVhiKcsTRIeMDqaVRtUU0/NIPakA8DApo5Yk05uFzSIPlpjBzgUjusMLO5wqKSTjoKG5YD8adnANIDnrfxXYzeILqzM6+THBC8bCN9xZmkBB46fKv5muhJwtVYLGOPUp78M5lnijiYEjaAhcjHv85/SrLdQKp26CFXgUn8dOpq9TSAU9KF6UjdqdQA08tSmkHOaVulAGDrPiyz0X7WrQT3RsLU3d55G39xFyQSSRydpwo5OK17K5+22Nvc+U8PnRLJ5cn3kyM4OO4rjtZ8A6jqba9FFrUUNprFzBcPG1qWcbAitGzbxlCqdAAfmPOODa1G70i48VWUI1yKK+0oGa6AvdmIwhyrRBsYOQSSMAAc9Kuyt/X9f8MDvf8Ar5G34h1tfDuhz6pLZ3N3DbjfMtvt3Ig5ZyGYZAHJxk+1P0jVTq0c0qWVxb26sBBNNtAuUIzvUAkhf94A+1ctpmlxeITqM+ka3HNoGoExXECzNcM5C8lZC5Ee7PzLg8bfunNLqFvaaN4KsvDev69Cb6SIW9rL9oNoWKqFVgA+444OMnLdB0AOVbddBX+47gnkUuait4zDBFE8jSsiBTI3ViB1PuakPSptqNbAKO9AoP3qQC1FPcRWtu807bI15Y4Jx+VS0g6UxmFpfii01DUb21My5iuhDBtjYFlMaNznvliO3AFb1VbaxjtLi7ljZy13MJnDHgEIqce2EH45q1Q7aCEHU0GgUGlbQA7UDqaz7mHV3+2/ZL20j3hPsnmWzN5RH39+HG/PbG3HvSXEOrv9t+yXtpHvCfZPMtmbyiPv78ON+e2NuPemgNE/doFZ13Dq7fbPsl7aR7wn2TzLZm8oj7+/Djfntjbj3q1Cl2t9cNPNC9swTyI1iKunB3bmyQ2TjGAMe9ICYfeNKelA+8aG6UAA6UhODk9AM04U3+KhsZgN4usRrq2fnp5BtmlLeW+7cGAx06YPpW+jBowy9CMiq7WMZ1MX+5/NWEwgZG3aWB/PirI+7TdrL+uour/roNP+sFO7U3/lqPpTj0pAIn3aRvvrSp9wUjffWgB56GmR/dp56GmR/d/GkAj/AHl+tPNNk6r9aeaYyOM9frRL/D9aI+rfWiX7o+tT0Ao6jrMWn3VtZpFJc3t1uMNvFjcVX7zkkgBRkZJ7kAZJqHw3rkPiPR01O1gmhhmZwizABiFYrngng4yPas7WvC2o6h4gm1PTdXSy8/TGsGDW5d0+YsHRtw2nJGeD0GMHkZN1p+m6ZoOkeEJ9ajg1NI4o7Vo7k27DaQPMCBsseOAc5PtkjRJNLv8A8P8A8Al3T8v+G/r5Ha6hM9vYyTRW0t06DIhiKhn9huIGfqRWX4b8Ux+JoIrix0+8js5IBILmYKqB84aLG7duXucbcgjORWFa+Idd8QakbOymsbKApJP5jQM8jQec0cbL84ALBWYEggYHBzipoLS88DaStnYXMd2l5qZFnDLGR5McjNLJubdk7V3nPsODQklv/XT+vQb8un/D/wBep2Cf65qJ/wDV1l+GdSm1fQrLULoRiS6hEpEQIXB5HBJxxjjJrVmGYzUtNXTGtbDo/uCoVOLg1LEcxiom4uBU9mND5/8AVmlhOYxSy/6s/SmW5/d0/tAMJxdVLOMwmopeLgGppOYj9KnowGW5/d1FJxcA062PBFNueJAaHrG4+pLOf3J+lV4JOMVLJ80X4VTi++amT1GkEzETZp7ndEfpTLgcg1IoBj/Co6jOD8K6xDpPhKXV9Vnvbh5b24iLPNJKBieRVHzHZGAABk7VHGTW3JqWorD/AGlcwW8VkFBSCOVXkfPQmRmWNevQFvZqo/DyQJoctnKskdxDeXLPHJGy4DXEhB5HOQc1tNoGnW9wZre3EIkz5kMZIicnuY/u598Z96uVru4luVbTxPDe3EdstlMhkONzXNswH4LKSfwBp19r8elXQhktZJiV3ZSeBMdukkint6YrQWxtkhZ4LaGKUKdsiwglTjrgdazdGsrl9Lt/+EiWG7vwGEkxt1TcNx2naCQp24yATzmp93cNSxY6omqxvJHA8Ow4w8sT5/79uw/PFUrrX7rQYlOvxQy27NtW5tDhifQwsdxPshcn0FWNYtL2JrD+wmitV+1KbqMW6kSxfxDdkbTjkHk5wPWrOm6PZWUpniiL3LDDXMzGSUj03Nkgew49qel7hrYyR4ugjmZfsM5wSM/abVc/g0wI+hANVL+Zz458PT2811FHf29y80DXTNGdqJt+QMUyMnlfzrom0+z88lrSAknJJjHNYGqsk3xB0CK1jcrZwXQl2xMEj3Km0ZxjnFEbX08/yYSudUhyv4VEp2uRT4jxTJBiSsih0w+UEUqHKUrDMdMhPajqMYDtkp8vK5FNlGJM1J96P8KQDYzlaZ92Q0sRwcUkw+bNLoA6QZjrgLXxS1h4s1m+ubfUJ9MN7BpiSo4aGB+AW2s2eZJAp2g/d5r0AcpiuU0zwk1tb21nf3iXFpa3bXm0RbWnlLlwznJ6M2cDuB9K1g4p3kTK7Wn9f0zpm+WSnvzHXNaVLp2p+Jb3UtH1NbmJY/JmjiuzKrPuzuK5ITGCB0zk9sZ6VOVqGraFDYmzxSP8slC/K5+tOlGRmpAeDkUVGjfLzRSA0xme4JPQVNO4SLaOpot02R571CT51zjsK6tkZk1vHtjyepqJiZrjA5AqaZ/Lj4/CmWyYXcepp+QvMklYRxHHXtTLZMLuPU0yU+ZME9KnY+XHn0FPd3AhlbzJgg6DrUxISPPoKit1yxc96LhssqDvR0uAtuuSXPeidtzKg/GpVGyOoYR5khc0dLATABE+lRQjdIzH8KdcPhAB1Jp8a7YwKOodBk78bR3qRAFQVCB5k59BUkrbYz+lPzEMT55i3btT5WwlEK7U+tMb55gvYUhkkYwgpv3pvYU9jtU0yIfKSe9PyAe52oaSNdqj1pspyyqPxqQ8CgQxvmlA7Cnk4BNMj5JY0SnjA70eYCxj5c+tI5yQBTxwoFMTmQmgB54FNj5yTQ5wtKgwooEJIeMetOHSm/ek+lOY4BNADV5Yn8KV/u0RjC0jcuBQA4cAU0cyH2p3Smp3PrTAcxwDQOAKRucD3p1ADSct9KUn5aavLE0rH5aBir92kPJFKOlIPvUAL2pE6UpPBoXpQAMegpaafvU6gBB1oNA6UHqKAFpP4jS00d6YhT0oHSg9KKQBn5qO1A+9QelAAOlIe1KOlH8X4UALSClpB0oAD2paQ/eFLQMQdTQ3SkXp+NDdqBDqaD8xp1NXqaBit92gfdpH+7SjpR1EN/5a/hTj9003/lr+FOb7poARPuCkb760qfcFNb/WLS6APJyKbGfl/GnnoaZF938aACTqv1pxpknVfrTzTGMj6t9aSb7v40Rfeb60s33Pxqeg+o8fdFRp/rWqQfdFRr/rzTYhZQNhPenJ9wfSkl/1Zoj/ANWPpT6h0I8YnwOKkl/1ZqNuLgVI/wDq2+lLowGwH5KZJxMpp9uflNMm4daT2H1JX5jNRWx+Uipuq/hUFtwzCjqgC44kQ1Pn93+FQ3XRT709WzF+FC3ArQtiQ0l0x4NMUkTEe9OuOUrPdFdRyOSmDVdP9afrUsJytRn5ZzUvYY6flBSwnKUsozHTYD8tHUCI8TGpJhmOmTfLKKlPMZ+lADIT8uKik+WanQH5iKW44YGp6D6jm+aKo4TUqcx/hUEfEhoAWbiTNP6x0k4+UGiI5Wl1AjiPzYpZhyCKb92U1JLylLoMF5Sok+WQ0+E5GKY/yyUAOmHygilQ5Wlb5ozUcR5o6gN+7JUkg3JTZhhgacPmjNLqA2JuMUx/lkoj+ViDTph0NADmG6OmRtzT4zlaiX5ZDQAsow+ak+8lNlGVoiPFHUCLOOKKWRfnNFRqPQ3J38qLA6ngUlrHtXcepqKQ+bcAdhVh28qH8K7Vq7mPQgmPmzBR0FWGIjiz6CoLVMne3WluXLOIxSXcAtlyS570XLbmCCplAji+gqCAebMXPansrAWEURx/hUEQ8yYuenan3D7UCjqadCm2MU+tgG3D4XaOpp8KbY+etQH97ccdBU8rbIye/ahb3Ah/1lx7CppW2x+9Mtlwu71pJTvlCCktEHUdAuFyeppsh3yhfSpj8q/QVDCuWLn1p+QeZMx2rnsBUcK5yx6midvl2jvUirtQCmtWIjmOcKO5qQDCiol+eYnsKkkbbGaPMCtcXdvZW815ezJDbwqWeRzgKKqWXiKw1W8urKzaYXNqiPNHLA8ZQPnbncByQCcdfWqXivSNR1TT9OXSlt5TbahDczW9zKY0mRCTt3BWxhtrdD92qGj6PrOmt4jvNSMU95f3JmiNpJhiioFRQXGFwB3z1JzVKyi2/wCtv8/wDW6S/rf+vmdig2qBWBD4x0e61Q2UMly1ylz9leH7HKHR8A5YFcqmGHznC8jmsmLVNUn8M+GFsrqSe+vsXJaYiMvGsZk2uQDgEmNCQO+as6Joms2njK61q5+xiLU7WMXcaOWMUkZbaE+UbhtYAscfd6c4FciTtL+v61JvpdHWMcKaSMYT3NYnh+5nv47+6mmaSGa/lW2U9EjQ+Xx7FkZv+BVu9BUbFDH5YLT+maYnLsfwpZDhaQBHzk+tEh6D1pVGFFNPMnsKBEmKj3BdzsQFUck9AKcxwpNNVA0RDgMGGCCODQMzV8QWUurXFiJ4B5MMUgk85fnLs4xj22frWqoworJTQLKPW570QQETQxRiPyQNhQuc599/6VrVTt0EN6yfSnE4XNNTufeh+lIBU+7SNywFOA4pBy5NADqavrQ3ANCj5RQBl6v4j0zRSy38zKY4TcSiONnMUQ6yNtB2r15PXBx0rQs7mK9sYLq3LGKeNZELKVJUjIyDyOvSuB1/wl4l1CXxNBaGxMOsTQbZ5J2WT7OqorwkbCFGBId2T9/pzkdBqc95beJvD9rHLNBDKZBL5W0wvtjY7Cp+YdMhs4G3HerUVb+u1xNu/wDX9f16mlrOtWWgafJqGqNJHaxkB5I4Xk2Z7ttBwPUngU/T9XtNV+0/YTI628nltI0LIjnGcoxADj3XIrmW0zWtWbVdPF4LrQ9RQxNPdOvmwgrhvLRUAZWBwMkYxkZBFTfYtZtfh9a6dd3v2TVordYY309gd7qmASXXpkbjwMDj6qysPqdcOlH8VR26yrbRLcOJJQgDuowGbHJx9aeo+Y1L3BaoU9KAOKRvu0o6UDA9QKSSSOKMvK6og6sxwBR1akkiSWMpMiuh6qwyD+FHmBnafrtpf3l7bpLCGtrgQriYEy5jR8gf8Dx36VpN0rN07QrSwu7y4SKEtc3AmXEIBixGiYB/4Bnt1rSPUU3aysIdSD7xpaavekMU9KB0FZ13Nq6C9+yWVpJsCfZPMuWXzSfv78IdmO2N2faluJtXT7b9ksrSTYE+yeZcsvmk/f34Q7Mdsbs+1FhF/wDipTWbczaur3v2SytJNgT7J5lyy+aT9/fhDsx2xuz7VZie8N7crPDClqoTyJElLO/HzblwAuDjGCc+1FhlhelDdqVR8tI33xQIdTV/iJ4pTSKMg56UDMuTxBZDWFsPPt8NAZvN89eCGA24/HPWtUYKgjkEcGsmTw/ZHWEv/It8CAw+V5C8ksDuz+GOlawAAAHAHQU9LL+uv+Quv9dhg/1ppzfcNNXmRqV/uGpGKn3BTW/1opyfdFNP+tH0o6B1JD0NRx/dP1qQ96ji+6frR1DoEuMr9af2qOYcr9ak7ChARx/eb60s33PxpI/vt9aWb/Vml0H1K19qtppiQC6dvMnbZDFGhd5GxkhVHJwBknoO9Q6NrFlrtqt/pchltpGZVcoV3FWKnggHqDWFrWjeIJvFDanoxsyn9kvaQPcSsGtpi24uFCkNuwg6jG3v0NGSy1Hwx4H0TTopZLe4he2tpZ7RlcIzOq7mDDLgs3IAGc1oknFd9Pzf/A+8luzf9dP6+47W/uY7OwmuJlkZI1LMIomkbHsqgk/gKztD8T6Xr3lrpUss6tbrcCUQOI9rYwN5G3dyMrncO4rJh8Wanq961lo2l2/WXNxcXRCqiSmLeFCHdkgkDIztPI61W0RL7wL4fNjcRQ3kT6kYrFYpCJHSWQvlvlwCoLHA4wpORQo9/l+QN6W7HZScTrUrDKmsrStTOsaNZai0PkfaYxIIw+7APQ5wM5GD0HWtX+H8KjVNpj3SZFb9/rSXP8JpLdvnI96W5+6PrU9B9SYcoPpUEHEzCpozmMVADtuDTfQB93/qx9aZE2Y6W6b91xUMBytS/iH0IjxcGpZRmKo5Ri4BqZxmL8KhDIrfpTJhiYGnQHDY96LnhgaXQrqSEZiNQwHDEVKnKVBHxKfrQ+hJyega448PnVfEGrO5mvZ4Io3SMAFZnRURUUMzEKOPmJxW3DrFw6NcSaZcQWKr/rHRmmc9tsKBmx9cEelYfglLLUfBclncC3uQL2782Bwr4zcyY3Kf61tW/h+3RWhFxcPYuuGspZPMjHoVJ+ZcY6A49q0lZSaYtQtdbtZ7xYY4r5Wc4Bk0+dF/FmQAfian1PVLeyZY5o7tmI3ZgsppVx9UUjPtVX+wNMsbj7RaWkSTRgsjMzYBx356VDZaf/b2i2154k0+3GobSriJn2YDHBXdg4IweRnmotForU1NN1CK/hZoEuF2nBE9tJDn6B1GfwrKk146dIf7ftGsEXrcq3m2/wBS4AKD3cKPc0kkM+iNZReH7O28u4u1S6WRn3CM8FkxkZHXnAwPXFTP4ft2uFm1OWTUplO5TckFEPbbGAFBHrjPvR7tr9A1JJfENiUIEeoOB0ZNNuGU+4ITBHuKy31G8j8baLFDe3H2HUIbh5LWWFVClFQrjKBweTkE1qN4b0iYu8lijO5LE7jyfzrBvTbRfETw1aWzxjyILsGNWyUGxMZHUU48rlp2f5Ml3OulGHBp55jpk4yoNOjOVrLqX0IouGxSzjoRSH5ZafKMx0raWGKhymKhHEhFPiPFNkGJKT1Qx8wyoNJEeKceUqKM4bFHUQjjbJXHReMIYfGGrwajez/YY7iCxtwLYtFHMVy26RVwuS6qNx6g12cw6GuQ03wpfPpsGnaobfyV1BtQupY3LNdSeaZFGCBtAO3OSfu4960p8t3zCle2n9f07HXRGklGGzWPZx3X/CU3MqX001j5W1oZNu1Zd3GzABwACDknJI7g1tyjMdZvRDAfMlMQ4bFLEeKa4xJS8wJWAJooByKKANK0Tjce9JcPvkEa/jU/EUP0qC2Xe5c966rdDPzLAAij+gqvAPMlZzT7uTACDqafCnlx0+oug25fCbR1NOhXZGPeoQfOuc9hU8z7IqL31Ahz5117Cp5n8uI1HbJhdx702c75Qg6CjZB1HWy7V3HqaSc75FQfjU2Qkf0FQQAvIXND0SQFj7kf0FQQDdIz065fChR1Jp8Q2Rj6U93YBs7fKB6mnxjagqH/AFlx7CppG2oTQu4iEfvLg56CppG2xmmW4wpY96SY7mCj8aFsHUfCMJz1NNkO6QLUmcLUUXzSM5p+QyY/Kv0qjcabZ6pGE1K0huo1cOqTRhwGHfB78mrczYjwOp4pYxtQCjqLoVprK0+0RXBtYftESGOKXyxvRTjKg9QDgcewqyFGzaRkYwQe9NPzTAdhT2bapPtQBWs7S2s0WCyt4reCMfJFCgRV78AcCrLnappsI+XJ70SHJCigBUGFFNb5pAKk6CmJy5NAD+gpkfQk96WQ4T60qDCgUdQEk+6B6mnDpTD80oHpTycA0xDRzITSscLSR9CaJOw9TR0AVOFFIeZAPSndBTV5djSAdnHWmoeD7mlc/KaEGFFMBJOgHvTqaeZBT80ANHLGqTaNphnubhtOtWlukKXDmFSZVIAIY45GABg+lXE7n3pX6D60DK2naXYaRa/Z9Ksbaxg3FjFbQrGuT1OFAGajv9G0vVLmCbUtNs7yW3O6GS4gWRoznOVJBxyB09Kv00csaLvcWmwtIvTPrSt92hfu0DEbp+NL2pCfmFLQAg+8aUng0i0N0o6AKvSkz81KKT+M0dAHU1KU/wBKRegoAG7fWnU1vvD60tAhB940MfloU9aHPy0DFXoKT/loKXtTf+Wn4UAO7UifdpT0pE+7R1EJIen1p2aY/VfrUnajqMjT77Usn3DSR/eb60sn3DS6AKn3B9KYT++H0p6fdH0ph/14+lMCQnimRfd/Gnt0P0pkX3D9aQugkx5X61IDxUc3VfrUmeKa3H0I0/1rUsx/dmmp/rmpbj/VGp6D6jozmMfSqE2nWM2sx3stnA93Eu1J2jBdR6A9R1P51dg/1dRSHFyKV7JMBI9OsrVjLa2cEMgiEQaOMKQgJIXgdASTj3pEt4LkL9phjlMZLJvQHaSCpIz04JH0Jqy5/dn6VBbNyab3DoRmCCzhht7WJIYYlCxxxqFVQOgAHSratmP8Kr3fUGpY/wDV0Xd3cOhBE374/Wn3R/dg1CnFwamuRmKoWw+o6B8xYqs5IuKltj8lRTcTUnsCJpeYTUVsetTtzDVe34Y4oe4xtzw4NTL80f4Uy7GMGnxH93R1GV4uJDmn3IyoNMzif8alm5h+lStg6iQnKVCeJ6ktzxzUc/yyil0BC/Z4YkdoYY0ZvvFVALfWkgPapusRqvCcOaHugCfhwalXmM/SmXA+UGliOUpdQIoztk/GnT8qDTDxNUsnMdLpYYkRyKreRCl4ZhEglPBfaN351NCaSYYfNHQB8nMdMiNSLyn4VDH8rkUPcAlGGzUg5Skm+6DREcil1GRRnaxp0wyoNNYbZKlcbo6AGxHK1Gfllp0J5xSTcNmkA9/mjpkRzxT15SolO16AKlppGnaS0n9l6fa2XmndJ9ngWPefU7QM9a0AdyYpsoygNERo3eoDF4fFLKOAaR/lkp5+aOl0AarfLRTASBiii4GxdyEkRr3NTxL5cXpVWEGSYu3QVm2fjDRdaur6x0m7ae509tl1GIJFMTc8HKjng8e1davqzJ9jUT99cFscCp7h9kPHU1Q8P6rZazp/2zTZHlhLFRI0LxhiDg43AZGe4qxKfNuAo6CjVLUNyW2TamT3qOc+bMEHapZpo7WEGWRYwSFUscZJ4A+prK0XXtM1nU9QtLC6Et1p0gjuotjKYmOcA5Az0PSiz2QXNkny4/oKr243uzmnXT8BB1NSQrsjFHUOgy5fChR1NSQrtjFV/wDXXJ9BViZvLjJpruHkQH97c+wqeRtkZNR2qYUseppLhsuEH40tlcOo63X5Sx70k5y6oOalUBI6hi+eYse1PpYCdflQCoYsvKWNPnfbHx1NEI2x/WjrYBZm2ofeljXagqKT55Qvapzwv0FPrcRE3zTAelSMdqk1HANxLe9LM2Ex3NHS4wiGcse5omOQFHeokuJRqAtfskvleT5n2rK+XuzjZjO7OOemMd6kzvm+lHQXUlAwoFRj5piewp7NhSabCPlz60wHSHahNJEMLTZTkhfU1IBgUuoEbnLhak6VGvzSk+lPc4U0AJHyzNRJwtEYwgpH5cCjoA9RhRTPvS/SpO1Rx8sTQIeTgGmx/dz60SHC/WnKMKKYDZOw9TT+gph5l+gpx4FIBo5djSucKaROVJ9aH+6PemMcv3RSH74FO7VGzFd7qhcqpwq4y3tzS2QD+1InIP1qG0uJLnT4Z5raS1kkjDNBKVLRkj7p2kjI9iamQYUU3uIH4UU4dKY3LAU+gY3rJ9BSnoaRfvE0OcKaABfu0N94Uq/dFIT84oAdTV+81Opqc8+9HUQrfdNC9BSP92nUDG/8tB9KdTRzIaU9DSEIn3fxofoPrSp92kfqv1oGL2po++afTF5Y0xDm+6aRfuih/uGlX7ooAY/3l+tSVG/MiipO1JDI4/vNSy/dpIu/1ol+7R0GOT7o+lMP/Hx+FSL90fSos/v/AMKGBK/Q/SmRfcpzn5T9KZAcpzR1F0EnP3frUo6VDcHG2pV+6KOodCJD/pB+tOuP9Uajj/4+DUs/+qNT0H1EtzmOoZv9eKltv9XUc/Eymj7IdSdv9UfpVe2+8frS2dzLdW8pms5rQpK8arKVJdQcBxtJ4PUZ59QKbb8SNTe6BD7z7qn3p8BzGKZd/wCrFOtz+6pfaDoV+l0anm5hNQPxc1Yk5hNSuoyG1Py0y54lFOteppLrqDR9ka3Jl5jqrF/riDVmM5j/AAqsDi4NJgS3X+rFJAcpSzjMVMtz2oe4LYjf5bipn5iNQ3HEgqZeY/wpLqBBD94ii56g0kfEhp9wMpmp6D6jozlKrjiY1NAflqKQYmoew0STDMdMgPFStzGaz7u5ubSxllsLNr64XGy3WRYy/P8Aebgcc0dRdCxMMSZqUfNHxTJxwD0NOiPy/hS6h0Io+HxTphlQaYeJalcZjpLYYkRytRN8stOhOKScYYGl0DqPYbozUcJxUqnMdQL8shHvQ97jHTD5gaenK0kwymaSE5FHUCMHbIaklGUzTJBtkzUnWOl5AMiPApsnyyUR8Maiv55IGt/KtZbgSyBHaMqBEME72yRxxjjJ56UdA8y1jdHUMZw1SRnIqNhtloAdMO9LGflpWGY6jjbFLqA1wQxxRUrLls0UrDNi3TZGK+azqOvaT4p+IGqaLbx3mn2mtJJqlrgiSWDMmdrA8D19vbIP0pcMEhII6jFZOn+F9CtxdPDounxG9BFyUtEXzweofA+br3r0IS5G/T9U/wBDB6r+uzX6nD+MfG+/wz4M1fwZqBhstR1e3tnWILgxNndGykcEYx7VmReJ9fsvH2o6ZrUt1HDdm7l0q8tWDQSpHE37ph/A6EZz1yMEEEGvQZfB3hdBHbW/hzSI4o5RKqJYRBVf+8Bt4Pv1rRTw3osckk7aVZmWUOJHMCnfv+/nj+Lv696fPF3SXf8AG1vuaEk1a/l+F/8AM8ObWdW8SaN8LLnVNXuzNfX032iRGVNzKxCtgDGQOKuQavN4f1n4l3dpqkenyrqNkizyRmQuCCCigfxMMgHtnPHUerr4T8P3NvBZtoenfY7aTzYLcWqCOJ/7yqBgGpb/AMKeG5obpZ9B0x/trB7jNpHmZgcgucfMQfWtPbRu3bTX80/0sTyOyTfb9f8AM8guvGniJPA3j65S/uYLnRdRRLJpGV5IEZ1Gxm5DYyRzn61614Zs9SstLku9W1qfVHvljmRJY1RbfKDKrtHTPPPP15Jkt/B3hw290j6DprpeFTcI1ohWUr93cCMHHatExRRCO2tY0ihjAVI41CqoHQADoKzlNcjst7fkv1VyrO/3/j/kT2qbUJI5NMuTukCCrA+SP6Cq8H7yUuay6WL8ywBsj+lQRDzJy/YVJcPtix3NFum2MZ7093YXQWdtsf1ogTbH9ailO+YKOgqwfkT6CjqBBJ88wUVOflX6VBDlpC3vT532x47mleyuAkA3OWp0xxHj1pYl2xio3+ecAdFp7IRLGu1AKjb55gOwqUnauaigBYlj3pvsMmPCk1FCvVvU0s7YTHcmnIMKKN2IbN90AdzT1GFH0qPO6cD0qUnC0ARD5pz6CpGOEJNRwjgn1pZjwB6mjoHUWIfLn1pJecCpFGFAqIfNN7CjoBKBxUa/NIT6VIxwpNMiHy59aOoDnO1DSRjCikl5wPU08DAoEMfl1FPqNfmlPtT2OFzQManLMaWQ4Q0R/cpJOcCgByjCimtzIBT6YvMhNDEP6Cmx9zSscKaEGFFMBH+7+NOApr/eUU6gBvWT6CnGmpyxpXOFNIYiDih+31pU+6KRuXUU+ghwpo5kPtTqavJJ96QDm6GkQfKKH+6aVRgCmA1+op9Mb/WLTqBjV+8aVvumkTv9aJPuUdBCqPkFI3+sWnL90Uw8yj6UgHkU2Pq31pxpkX3T9aYCyfcNOX7opkvCU8fdFHUZGf8AXD6VIfumof8Al4/CpW+6fpSQDIeQfrST8L+NEH3T9aLj7o+tLoPqSKPlH0qLrcmpV+4PpUK/8fBpvYETSD5TUdv92pJPumo7f7tHUQ256rUy/cFQ3P8ADUy/cFJbh0K6D/ST9amnH7o1Ch/0k1NN/qm+lC2Y+oy2/wBWajuf9YtPtvun6025++tL7IdScf6uqsPExq0vMYqpH/x8Gh9ARNdD91SW3MdLc8w021Py0faDoRT8XAqwf9VVe5/1oqwOYvwpLdjK1uf3h+tPvB8oNRxcSmpbrmMVK+EAg/1f4VXcYuKmtzlBUU/EwpPYZNIMw8VDb8E/Wp+sNVoT+9NN7gh10OlPj+7xSXQ+QUkByopdQ6EI4mNTTDMNQy8TVOeYqSGQwGkuBhwaSE4YinXA+UGp6B1HrylV04lNTwnK1A42zUPoMlmXMeabCeKkPMVQRH5iKHuHQJRiSpRzHTJx0NLEcrS6h0Ik4cinzDKimN8s1SsMx0dLDGRH5ajkGJadCetLP1BqegDyMxmoYjhsVLGcpUP3ZKbAfMOAaWLlaWQZjqOE84o6gNPyy1KwzHTJhyDT0OUpLcCOM4NLMuMNTR8slSSDdHS6B1CM5WosbZDTojSTDDA0dA6kw6UUxWytFVdAaM7Ga4CDoKs8RQ59BVe0TcxdqdeSfKEXqa6Ftcz62GWy+ZKXb1qa7k2x7R1NOgTZGKrufOucDoKWyDqT2ybI8+tQyt5twF7CrLsIoT7Cq9quWLnvTfYEWSRHH9BVa3G+UufWnXb4AUd6lgXbGPWnu7C6DLl8R7fWnW6bIxUD/vbkAdBVp2EcZPoKFq2wK0p824CjoKsk7I/pVe1Xcxc0+6fChR3oW1x+Q23G9yxqS4fEePWlhXZGBULnzLgDsKNkLqTwrtjHvUL/ALycAdqsMdqfSoLcbmLGm+wicnap9qigGWZ/WluGwmO5p8S7YxR1AbO2Ex6mnxjagqGQ77gKOgqcnC0Le4dCFvnnA7CpScLVKG2lbVDefa5fK8ox/ZcL5e7Od+cbs44649qtTtiP601tcfWwkIJJb1pZiQmPWnRjCAVG/wA0wHpS2QiVBtQVG3zTAelS5wtRRcsWp+QEpOBmo4hkFvWlmbEdLGMRin1ASU/Jgd6cowopjfNKBUnal1AjJ3TAelSE4FRx8ux96dIcIaOgCRDgn1NEp+XHrTk4UUxjmRRR0AkUYUCmdZPoKeKZGcsx96AH5602PoTRIcIaVBhRR1ASQ/Lj1NOHAFMfmRRTpFLRsquUJUgMuMr780AN6y/Snk1V0+CS2tY4Z7mS6kjQI08oUNIQPvHaAMn2Aqy5wppiEj+7miT7v40qfdFNfllFLoMkHQUzrL9BT6YvLsaBDieKbH92lc4U0J90U+oCSfd/GnDpTH5ZR71JR1AZ1l/CnHpTEP7xqc5whpdAGx/dolPyj60sf+rFNk6qKOg1uSDoKjB/fGpaiTmVqGBI33TTIfu/jTnP7s/Smw/cp9QEm+6PrUg+6Kjm6r9akHSkHQiHNwakc/uz9KjX/XmnvxGfpQge42D7v40246L9adB9ymz/AMP1pfZH1Jl+4KgB/wBINTr90VXX/j5ND6CJn+4ajtvu1JIf3ZqO3+6aOoCXP8NTL91ahuf4amT7i0L4g6Fdf+Pqppv9U30qBf8Aj5qxL/qj9KUdmMhtT8ppt195ada9DTbs8g+9L7I+pOn+rFVU4uDTtNt5ba3kWa7lui8rSK0oUFFJyEG0DgdBnn1JpvS6olsgXUsXH+pqG1PFTzcwmq9q3Wh/EHQS6GGBqeM5j/CobztUkJzGPpQviDoVl4uDU8/MNV24uKsyDMP4Uo7MZDbGmXI+cGltjgkUt2Ohqfsh1JU/1dVV4nP1qzCcx1Vb5Z/xoeyGTzjMNR254qV+YaggPzGh7iWwXAw4qVOY8VHc9AadCcpU9RkC8S1LMMx1C/E1TtzFSWwEcDU2cYkBoiOGxUGs2Z1DT5LYXVxaGQD99bOFkXnPBIP8qN0PqXE5j/Cq68SGpoD8gGe1QycS0n0AlmGY6ZCeKkPMf4VDEcNijqMJ+HBqReUplwOM0sJyKXUXQjX5ZKkmGUqOT5Zal6x0LawxkJpkww+aIzhsU6YfLml0DqPX5kqBTtkqWE/LUcgxJR5jHyjKZpIWp/3o8VSmgkmlgMV1LbiKUO6xhSJRgjY2QeOc8YPHWjqLoWJRh8ipF5jxTZhlc0RNwKOodCNflkIqSYZTNMkGJM1IPmjoAiVvloph4OKKQ7HQRL5cX4VVUedclj0FT3cmyPaOp4otI9qZNdW7sY30uOuH8uE+vaorSPgs3em3L+ZMEFWlAji/CnvIOhXu3ywjHep4lCRj6VViHm3BY81ZnfZERQu4eRXH766PoKsyt5cRPtiorSPCknvTbt9zBBRsrj6i2idXNOu36IO9TRrsjH0qqP310T2FJ3SsHUsRJtjFQN+9uvYVZlbZEfpUFqnVj3pvsInc7I6htl3MXNF0/AQdzUsS7YxRuw6DLlsKFHU0+FdsYqBj5k4HpVljtTPtTW9wIH+e4C9hU5+Vc+lQW67mL0+4bCY9aOlw6jYBuYsafO2I8dzSwriMVHKd0yrRsgJYl2xio5PnmC+lTdBUEPzSM3vQ+wE54FQwjdIzU+ZtsZpIFxH9afUXQWU7UOKWIYQVHMcsFqYDAo6h0IpeZFWpf4ahT5pyfSpXO1CaAI4/mlJp8h2xmkhHyZ9aSY5wvqaOgdRYhiMUkvLKKkAwtRfen+lALclGKjUZmJ9KkbhTUcPQn1NAD2OFJpsQ+SiY/J9acvCgUdQGynoPU08Co25mA9KkNMBg5mPtT24UmmR8sTSynCGl0ASIfLn1NEn3cetOQYQfSmycuoo6B1HgYFM6zfSpKiTmVqAJD0pkXQ/WnOcIaSP7lPqAkv3Keo+UVHNyVHvUg4FLqAw8zCn1EvM5qRvun6UAMi5LH3p0vEZpsA+X8aJj8v40dB9R6DCD6Ux+ZVFSL92oicz/AEo6ASmoovvsalPSo4ecn3o6i6DpD+7NEQ+QUk3EdOj+4PpR1DoRy/eWpe1RSf61am7ULcCCP/XNUkv+rNRw/wCsanzf6o0lsD3CD/V1HP8AeWpIP9WKjuPvrQ/hH1Jx0quv/HwasDpVdP8Aj4ND6AiWT/VmmW33afL/AKo0y2+7R1DoNuuq1On+rWobnqtSx/6sUL4hdCuOLqrEn+rP0qsf+PqrMn+rP0qY7DILTvReDhaS1+8adefcFH2R9R8H+rH0qu3FzVi3/wBWKrS8XIolsgW5ZkGYPwqtbcMatN/qT9KqW5/eGh7oFsPvB8oNPtzmMUl1zHTbY/LS+0PoRTDbcfjVg8xfhVe44mBqwpzH+FJbsGVYDiQ1JdD92DUUfExqa4GYqS2DqNtzlahm4nzUtseKZcjD5pfZDqTDmI1WjOJDVmM5jqr0mP1oeyGia4GY6ZAflqWXmGoIDS6h0Gz8Sg1MnzJUdyOhp8PKUdQ6EKfLKallGY6hb5ZqnPMRpLsMihNNnGGzRCfmx7065GRml0DqOTlfwqAfLLU0JytRSDbLQ9hksgzHUcJqXrGagjOHNJ7ghZxyCKkj5WmzDKcUkJo6h0IyNstTMMx1FMMODUqcp+FC3AiiODiicd6aPlkNSyjdH9KXQBIjkc1Gw2yUsJpZxzml0AfjdGaiiOGqWM5Woj8slNgOmXK5pYjxSsN0ZqOI4NHUfQbIvzmipmXJopBcuufOucdhVxiIofwqtZxknce9LeSZwg610rRXMvIbbJvkLtU13JtjCjqadbp5cYqtIfPuuOgo2iHUntU2x5NR3DGSYIPxqy2I4ifQVVtl3ylz603tygu5bUCOP6VUhHnXBY+tS3UmyPHc0tom2PJ6mm9XYOg64fy4jTLSP5Cx71HctvlVB261aQbIx9KN2LoV7piSEH41PGNkYFVU/e3Rb0qzM2yMmhdwfYg/1t17A1ZkbYh+lQWi8FjS3bfKEHektI3DqFsuSXp1y+EwOpp8K7IxUD/vbjHYU9kHUngXbHUMh8ycL6VYY7UPsKrwDdIXofRAWPur9KhhG+RmPSnzvtj9zRANsf1qt2LoQTahapefYPtEf2sxecINw3+XnG7HXGeM1PAuIx70yc7nVamA2qBSWrAhnOXVR61OOFAqup33BPpU7napNC7jIl+ecn0qVzhSfaordeCx70s7YUD1NPoAQD5c+ppZz8oHrTkG1BUb/POBR0AlQYUCom+a4HtU3QVDD8zs3vQImJxUUIyzNT5W2oaSEYjFHUOgTHEZpYxiMUyY5ZV96lHC4o6h0I5DmRR2qWoV+ac+gqRjhTQDGJ80rGnucIfpTIeVJ9TSzHCfWjoHUWL/AFeaZcSLHGWkYKqjczE4AA71IgwoqKQ5mUUPYfULO7t76yhurKZJ7eZA8UsbBldSMggjqKcDmc/SpO1Qx/NIxoZJMThSaih7n1p8hxGaSIYjFHUfQJ+I6cg+QfSo5uSo96lHC0dQ6ET8yqPSpe1RDm4PtUrHCmhARRjLsafJxGfpTYehPvSzHEdHQOoRf6sUkv3lHvT4+EFRyf61aHsHUlA4qJRmc1NUMXMjH3oGiVuENRw/dP1p0hxGabD/AKsU+ougTn5QKen3RUU/b61Kv3RS6h0Im/19Tdqg/wCXmpj900l1Aig++1On/wBSabb/AMX1pbg/uqX2R9R0I/dio5+ZFqSH/Viop/8AXLTfwh1LHb8KrR/8fBqz2qtF/rzSe6BE03+qNMtvumnzf6s/So7X7pp/aDoJddVqaL/VrUF11Wpov9WKS+IOhWY/6V+NWX5jP0qsx/0o1Zf/AFZ+lEeoMr2v+sNPvPuCo7Y/vDUl2fkpfZH1ItLvra/hdrO4jnWKRoXMbBtrqcMpx3B7Uk/FwKmtT8lQ3P8ArhSeyBbstZzF+FUof9cauJ/q/wAKpocXB+tEugInuRmKmWp+WpLj/U1Da9DTfxDWw26++Knj5jqG77fWpYD+7FSviDoVRxcH61ZlGYfwqu5xcfjVhuYfwpLqBXticmluhTIDiQ1Lc8pS6D6iwHKVXk4n5qW3PFR3AxJmk9g6k55jqtCcSGrCHMdVl4mND6AS3AymaS3Py06XmOorc0uo+gkwxJmo77UbTStLlvdRmEFtEAXkYEhcnHb3NTXA6GnQtlaFuD2IEOJOOlTSjMdQyfLNU/3ozSQEUBxRcD5gabGcOafMMpml9kfUdGcrUDfLJT4TTZxhgaXQFuStzHUMRw2KmQ5X8Kg6SGmwJJh8uaITkUr8x1HEcNR1ASQbZKlHMdMnHQ0sRytLqBGvyyGmXt5bWogW6uI4muJBFCHYAyOQSFHqcA8e1PkG2SpRylCAZCfWkmGGBpqHDkGpJVylLoHUVDuWoektPhNJMMMDQMmByOtFRq3yiimI2owI4fTAqomZrgk9M1NeSbI9o70Wke1cmujd2M/Mknfy4TjriobOPJLnvTbp98oQc1aiUJF+FPdhsiG8kwAg71LbJsiFVR+/uvYVblYRQk+1Nbti8irKfNuNvUCrnEcfpgVUtE3OXNS3kmE29zSjtdj8iKAGScsasXL7IeOtNtU2xZ71DcnzJggo2VxdSW0XClscmm3bZZUH41YQbIx7Cqkf765J9DTeiSDdlqNdkYqsf3tz7CrMzBIiagtV6tQ+iAsOdkZPoKgthuYsaW7fCbfWpIE2xij7QbIS5bamB1NLAu2MepqCY75woq1wqfQULe4Fedt0oUVYAworNu9RtdPQ3V7L5ce4IuFLM7E4Cqo5JJ6Ac0adrtjrMN2dPeRjaTG3mDxMhSQAEryOSMjpTjdpsGWk+e4JPQVYkbahNRWwwpPrWHqXjHSLO+udPlkuftdvs3QLZys7hyQCgC/OPlOSuQMHJpxTtoLzNy3X5Sx7mlnb5cetPjXbGBUL/POF9KXQfUnjG2MCoX+acLU/aoIfmmZjQ+wifOKgh+aRmqWVtsZpkAxHn1oe4D5TiM02AYjFJcHgL608FUj3MQoAySTjFNbh0GXB6L61KvCgVipr9lPr89gJ4AIYYpRJ5wO4uXGMe2z9a2jwtMCHO64+lTE4Wq8A3OzH1qWU7YzS6XGMg5JPrT5jiP60kAxEKZOfuqPWi+gdSWIYjFRzHLqtSqMKBUJG64+lJ7B5k46VCvzTk+lZereLdH0VpxezuPs+zzjFC8giLnCKxUHBYkYHXketakGCWJ9appiJWOENRwD5SfeodVv4dM0ya8uBKYoV3P5MTStj2VQSfwFU9D8RabrfyaZLJOvkpMJhC4jZXGQA5G0nHUA5Heizb0DZGnOfk/Gnpwg+lRTcsoqYdKS3AifmZRUuahBzcHPapTgKaEBFFzKxqSQ4jNMg5BPvSXUscNuzyuqIOrMcAUug+o+EYjps3QfWs/S9ctL64u7dZYQ1tOIVxMCZcxo+QP8AgeO/Sr8p/eIKb0QdSYfdFQnm4HtU1QpzOaGC2Jui1FB95j71I5wh+lR2/wB0/Wh7i6Dpv9WaIuIxTZzhKfH9wULcOhFOfnWpx0FV5v8AXKKsDpRHcGV05uTU7/cP0qCLmcmppiBGx6cUlsNkVt0P1pbn/V/jWXb6/ZLqwsTPb4aAzeb568EMBtx+OetadwQUUjBB7+tFny3DrYli/wBWKgmP+kLU8X+rH0qvKf8ASBQ9g6lrtVWHmc1aP3aqw/684oluhLYmn/1ZplqflNPuP9SaZa/dND+IfQbdnkVNEf3YqC76ipoceWKS+IOhWf8A4+vxq23+qP0qpJ/x8/jVs/6r8KI9RsqWx/eGpbv/AFeayLvXdP0ifZeSsH2NKUjjZysa/echQcKPU1fhv7fVNHt7+zYtb3MSyxMylSysMg4PI4Pekk+S4PcntD8tRXX+tFLbuI4i7ZIUEnapJ/ADk1iW/i3SNYuoIdOknleVpFINrIvlFGIYSZUeWcqcBsE44pWbjoGzOjj5i/CqXS4P1q3Ccx/hVRv+PiiWyBFqTmE/Sq1qfmNWW/1P4VUg/wBYcUPcESXZ+UU63OVpLofu6S2+7S+0PdEM/E9WV5i/Cq9zxIDU8X+rpLcZVj4mNTz8xVX6T1ZfmI/SpQPchtzikueoNJAcMadcj5QaPsh1HwnKVXk4mqWA/LUU4xIDSeyH1JzzHVeI4kIrmvC+usvhx9U8Q6s7ma8ngijkRAAVmdFRFRQzMQo4+YnFaEer3Bla4fTbiGyUffkRmmf02woGbH1wR6VUotMSZtT8x5psB4qimt2l0yQxxX4ZzgGTTp0X8WZAB+Jps2rW9jMI5o7tmIyDBZyzD80Uj8KTTuHQuzjD5qSPlKpx30N/Ez26XChTg+fbSQkn2DqCfwqiniA6eh/t+0awRet0rebb/UuACg93Cj3NCT5rDvoafSapn+aOsabxBYiU7Y79xnhk064ZT7ghMEe4qncajeR+M9Figvp/sOowXDvaywooUoqFcZQODycgmlGLbsJtG/EcNin3AyoNRrxIalkGYqhbFDYTlajlGJKdCeaWcd6Ogx6/MlQKcSfjUsR4qKTCyUPYRLIMx1HCealHzJUC/K9D3uA+Ycg06I5WiUbo64nwp4whutUure/vZ5DeahNDp2+3Ij2R8bBIFCliUdsE5xVRi5N2E2kjsX+WSpesdMnXkGnxHK1HWw+hEhxJUkoymajb5ZKlHMdC7AQKxxRSHg9aKVyjUkPnXOOoBq6cRQ1Us0y240+9kwoQdTXUnZXMXuR26mSYuas3Unlw4B60lpHtjFQXL+bOEHantEOpJZIQCxpLx8kIKsIPLi+lU0zPdEntQ9Fyh1uW4ECRiqsh866wOQKtzN5cJNVrNNzFjQ97Au5aJ8uP8KqWy+ZMXNTXkm1No6mltU2x03rKwdB1w+yE+pqO0TCbj1NR3bb5QgNW4wEjHbFG8hdCvdtnCCp4U2xiqgbzbr6VbkbZETntQnuwfYqufNusDoKuE7Uz7VUtRuct71Ncvtjx3NJbXDqRW43zM5qedtsRptquI8+tR3LZdUFPaIdTn9Z0fVrzXtD1DTVtZYbF5mliuZGQBnTasgwp3FQW44+91Fc2brXPA/hmK1uFs7nVLiS8vrlhucPyWB/h5LvGoHvj3r0yMbUAqsf3lz9KvmtFJhbW5x0+t3OreHmsbqSCC5vNXXTovs7lTtUqZec5yAsoyPQcc1MNI1658a6dr06WMMiwSWdzAkrOEiLKysrbAWbKng4A3d8c9o3yIccd6gtxucsabla1l/VrE2/r53/y+4yNCnW/8Q67fxzeZDFLHZRhXyv7tNznHQHdIQf9wVsQ/PKze9STNtjPvxSW64j+tS9WkNDpW2xGm264jyetNuT0WpoxhAKS1YEVw3AX1NSoMIB7VXf57jHpVnovNC3bArMd9yB2FWGVSjK4DKRggjg1BB88rNUszbYzQtmxmPZ6DZprlzfiCAiWGOIR+SPkKM53Z99/6VsSttjNMtlwh96Lk/KB70XdhdR1uMR5ptyeAPepYxhBUEnzTgelD0Q1uToMIBUDfNcgelWKrw/NMzUPsJFjoKgg+aRmNSyNhGpluPlJ9ab3Doeey+EvErwLazHTzb/2+2o3MnnuWuo95ePcNnG0iMbcnO3qO738cavHeSrb29l9m80pHJIrA4NysEZJ3fxES/gufY9/cH93j1p6L8gzVqXvK60X/AX6Caun3/4Lf6nDy3C6v4skuvCWoWt0Y0a1v1OoM6puPD+WNwyhQgD5d25ueOXafb2/gzwKdJ8U3tnBZJNJBZsLswNMhdmQM/y7TtIBAOOCc84rsZTmZVqfpU82lv6/rcLa3MjQrRrLRrC1e6e8aKBR57sWL8dcnk/U81r9BzUCc3BNTOcKaV7tsdrWRFD80jGpJeI2NMt/uk+9LcHEdLoPqEA/d1HdxpLF5cqK6MeVYZBqaIYjGPSop+ZFFHQOpS03QrSwury4WKEtcziZcQgGLEaJgH/gGe3Wrrc3AHpU/Sq683Jpy6ISLBqGH/WMalbIU/SobfvQ9wWxTvptXSO++y2dpJsCfZPMuWXzSfv7/kOzHbG7PtTTNq6x3v2SytJNgT7J5lyy+aT9/f8AIduO2N2fatGc/u6WEYjFO/vB0MzUZtXX7YLWztZNgT7JvuWXzSfv7/kOzHbG7PtVy2a8N1cLPDElqoTyJFlLO/HzblwAuDjGCc+1OuD86ip1+7+FJPUHsQPzcirB6VW63VWD0P0oQMgg5mapZ/8AUtn0qK2++xqS4OIzUr4RvcyYPD9kdWW+8i3wsBh8ryBySwO7P4Y6VpXIwqgcAdBT7f8A1dMus5FNt8tg+1cmiH7sfSq8n/HxViP/AFY+lVn/AOPmk9kNblo/dqtB/rTVk/dP0qrb/wCtNEt0JE9x/qjTLT7tPuP9SajtPu038QdBt31FTQf6sVDedRUtv/qxSXxB0K0v/H1+NXD/AKr8Kpy/8fNW/wDll+FJbsb6HnGueGfEl/feJ10x7IJq1vFBDcSzMskcaja8eNpAB3Od2f4uncWda8U6ho1ydPtLezFpZIsUjFWOzbbtLJj5h8qqIv8AvrHfI7GH/XmrNwAYTnkYqlL3bMVle55/Z+MNfnvLS3sbPTpZZZI4iksjRq7/AGfzZdrjdgKWUA7T1x7h8tnrHhCLxBej7JcrfTpc26jO5rmQJGU28YXeBg5zz7V21rjPTpS3nanzq10gs76md4V1O41bTJp7rycpdTQp5OcFUYrzknnIP14OBVyXif8AGrFtxGMcD2qvPxOKzk7pDjuWjzD+FU4T+9NWx/qqppxMamXQaLNwP3VQ2x4qaYZhqvbnBofxAthbofMKkgPyU27+7mktz8tL7Q+hDJxPVnGY8e1V5+JasJzH+FJbjKsRxKamn5iqAfLMasScxULYTIrc0lyPmpsBwxqS45UUug+pyXgBbLUfBstlcrBdKL2782BwHxm5kxlT/WtL/hH4IZTCs9w1i4w1nK+9B6FSfmXGOgOPata0ghhZmiiRGY/MVUAn60TjD1UpX1Qkiinh/TLRTcWlnEk8YLIzM2Acd+elZumWA1vT7e78R2FuL8qyuIi+zAY4K5wcEYPIzzXTRnMdVuk1JyY0jJv7ebRRZR+H7O28u4ulW6R2fcIzwWTGRkdecDA9cVLbaDbySrLqckmpTKdym5IKIe22MYUEeuM+9bDcxVBEcNQ5O4rGdd+GtIkkaR7JCzMWY5PJP41kXjW0fxB8M2ls8eYLe7BiVslBsTGR1FdZcDKcVDBBEJvOESeaeC+0bvzojJqWv9aBa6BxtlqfqlRzjDA0+M5Ws1uUQocSYqWUZSomG2SpusdC7AyOE0k68g01DiTFSzDMdHQBIjlajcbZKWE8YomHOaXQYl2k02mzpaOqXDRsImforY4J9s1xlpocmj6XpkupiGK18O2zyokbmQyyiMqZG4HGC5x1Jbtiu3iORUT/ACyfjVKTS0/r+risnuclpPifWL7WLax1C3s4vMdlkCBs/JCrPjLfwu4X8/TJ66I44qXG5PeoE4kxTm02mlYUU0rMfMOc06I5WiQZTPtTIjjip6lDXT5zRUrLz1opWC5sQgRxZPpVPPnXJx0BqxdyBYdo61Bar3PWul7pGXmXHYRQGqdsN8u40t3LkhKktl2qKHqw2Q+8l2R7R1qOzG3k1XnYyz4q4gEcX4Ur3dw6EV5NuYIKnt/lj/CqK/vJ81ckby4fwpp9RvsVpWMtxgdqu/6uHPtVO2Tc5NTXkmIwoojtcT7EMOZJyTVq4fy4qjtI8Lk1HdvukCijVRuHUfaJzuNLevgBR3qa3XZHVRz5tz9DQ9I2Ddlm2XbHUFw2+YKO1WshIfwqnAPMmLH1pvsCLy/JF9BVNCZbkn0NWZ32QmobNerGm97AWJW2Rk1DajcSxovHwoUdzT7ZdsXFG8hdBbltsf1NFuuI8+tc1pfiK81nULq3m06K2Swnkt7mQXW/bIuCu0bBkFGDEnGM4wetdJazw3NnFPbSpLDIgdJEbKsD0IPcU7PmDyGXL5YIPWp0G1AKqr+8ufxq052xn6Uo9WHkVifMufpVpvlQn2qtbDMhapbhtsf1oW1we5HbjdKWNTTttjNMthhM026PQDrRtEOo62GFzSXR4CjvUsQxGKry/PcACm/hBFiIYjH0qGU7rgLU/RfpVaIbrgk0PawItfdFV4juuCankOIyfaobUdWoe9hdCaQ4QmorYcE+tLcHEePWnQDEYo+0A25OI/qadCMRio7g5ZVqdOFFC1YdCCc5kVanAwPwqsTuuvpVDxNrsnh6yt7wWYuYHuI4JcTbGTewRSARhvmIzkjA5px1YPY0PvXR9qsNwuaybDU08+C21OS1tdSuELC0S4Dk4znaSFLDAznHr6VqSnEZ+lGyAit+WY1JOcRGm2w+TNFyf3ePU0vsj6iwDEYptyfuj3qWP7gqCbmZRQ9g3ZYUYUfSoG5uAKn7VXTm5pvoJFk1Wh5mY1OxwpqG35JNJ72BbE0pxGajt/u0twcR0W/EdHUOglz9ynxf6sVFcnoKmT7g+lC1kPoV5/8AXAVZHT8KrSc3IqyTxQgZVTm5NWXO1D9KrQ83BNWJT+7P0pR2Ahte9Puf9XTbX7pouvuUfZDqOt/9XUd198VLB/q6huT+8FH2Q6llPuD6VUb/AI+qtp9wVUP/AB9fjQ9kCLbfcP0qrb/6w1af7jfSqlt/rTQ90CJ7n/VGmWn3afdf6o1HadKPth0EvOq1Nb/6sVBedqltz+7FC+IHsV5v+Pj8atj/AFR+lUp/+Pn8auL/AKqkt2NlOI/6Qasz/wCpNVU/4+DSa3dz2OhXV1Z26XM0MbSLC8vlh8DJG7Bx+VKPwsN2PtT8xp950FZPhnVJdY09L9rZYLe4VZLb97vZ42UEMwwNp56ZP17DWu+UoacVZgmm00Lan5BUFz/raltj8gqK64kFS/hGtyxGcx/hVQcTmrURzH+FVH/4+DRLoCLcnMP4VVgPzkVZP+pP0qpF/rDSlugJrr/V022+7T7jmGobc0P4h9AuRhqliPyfhUdz0p0PKiktJB0IH4mqz1jP0qtPxLVhOY/woW4PYrR8Sc1NMMx1D0mqd+YjSQyGA4NLcimxH5qkuPuUlsHUSE5WopRiQU+E/LjpXPS6zd2979mQR6uFJVpbVSGhI/56Yyv1wQ3ohp2ug2Z0i8pj2quPlkxWXp2uXlzdxQy6d5SOcF9txx/31Ao/MitSTiWiSa3BO5M4zHUUR+aphzGfpVdeJDUve4yWcfLSQtxTpOY6ihPNHUOgTDDVJGcrTZxwKITxS+0BG3yyVN95DUMwxJmpk5WhbgQJ8rkVLKMpUR+WSpzzHQtrB5kULc4onHOabGcOa5vxDcT3Xiu20ozzQWv9mXF0WikKFpAyKvI/uhicepHpQtVb+tFcf9fodTEflqJ/lkrK8Hajcat4Q0q/vf8AX3FpHJIcYyxUZP49a1px3pzTi2n0Ji7q/ckHKVAvyyVNCcrUcg2yVJRMORRSK3y0UEWJZ38ybb71cjGyLNcJ4l1nXtP1iOPR7jTooWhDMLqzeZt25hwVlQYwBxj15rLm8V+MhHg32h49tLm/+Sa6oQb94ly6HoY/ez57Zq658uGvLLfxR4wB+W90P8dLm/8AkilufFnjE4DX2h49tLm/+SKaptK9xOSPSLdd8hJqxcvtiwK8yt/FHjFR8t9of46XN/8AJNRT+K/GLPhr7RPw0ub/AOSKfsmo7hzK56faJ3NF4+WCivOIvFXjJU+W+0L/AMFU3/yTVc+KvGDzZN9omc/9Aub/AOSKPZNRtcOZXPVbVNqVXnbfcYrzz/hLPGaxcX2hY/7BU3/yTVaLxT4waTP23Q8++lzf/JFN0nZK4KR6ynyQ59qpJ+9uM153N4t8ZiLBvtCx7aVN/wDJNRW/inxiDlb3Q/x0ub/5Jpum3pcSktz1aVvLgNVLVd0hPvXm9z4s8ZFcNfaHj20uYf8AtzSweKfGKr8t9oX46VN/8k0ezbkHNoen3b7Y8Cm2aYXJry+48V+MnkAa+0P8NLm/+Sanj8V+M1j+W+0L/wAFU3/yTTVNuV7i5tD0a8bLBBViBdkYryhvFfjF5+b7Q8/9gub/AOSatHxZ40WPi+0L/wAFU3/yTQqbve4c2h6HK3mXOOwqxPAs9jLBI0iJIhUtFIyMAR2ZSCD7g5ryuLxV4xMuft2h599Lm/8Akmp5fFvjMRn/AE7Qse2lTf8AyTSjTeruPmOrbw9Hbw6m+gxrDfXkOwPLM+3cF2hz15wBk4ycDNdBFBHY6bFbQrtjgjWNAOwAwP5V5hb+KvGQyVvtD/HSpv8A5Jp1x4s8ZlMG+0LHtpU3/wAk1ShLl3E2rnpdoMsWqW6bEWPWvMIPFfjNU+W+0L8dKm/+SaZceLPGTOA19of4aVN/8k0vZtR3HzK56lbLiPNR3TbmVRXnCeLPGYUYvtB6f9Aqb/5JqBvFfjIz832h5/7BU3/yTQ6bsHNqerRDbGKrOd9wPavOT4t8aBf+P7Qf/BVN/wDJNQR+K/GRlJ+3aHn30qb/AOSaHTdrXFzHrHRfpVWL55y1ebyeLfGgQ/6doP4aVN/8k1HB4r8ZDJW+0L8dKm/+SabpttIFJHpMFn9ikvZjPLKbqXzSruSsfyhQqgngYXPHck96ltR1NeZz+LfGZjwb7Qse2lTf/JNEPivxmsfy32hfjpU3/wAk0ezbe4cyPT7hsR/WlgGIxXlk/izxmxAa+0L8NKm/+SamXxZ4zCgC+0H/AMFU3/yTQqb5tw5lY9JuTllHvU6DCD6V5S/ivxk03N9oWf8AsFTf/JNT/wDCW+NMf8f2g/8Agqm/+SaPZu+4cx6O3z3WPSrJ4FeUJ4r8ZecT9u0LP/YKm/8Akmpn8W+NNp/07Qf/AAVTf/JNCpvuHMekQ/NMWqnrOg6frEcg1JJZUeIxeX57qoHXcFBwG/2hz715/B4r8ZDO2+0L8dKm/wDkmlm8W+MymDfaFj20qb/5JpqDSDmVzuLWxun1iFpre3+wWa5tnFwzS7yu0kqUx91mGdx6ng5417g/u68wh8V+MwgxfaF+OlTf/JNJP4s8ZnGb7Qvw0qb/AOSaOR2FfU9ShGIxUVwcuorzZfFnjTaP9O0H/wAFU3/yTUL+LPGRmGb7Qs/9gqb/AOSaTpu24+bU9WXhRVcndc15x/wlvjTH/H9oP/gqm/8AkmoE8V+MvOJ+3aFn/sFTf/JNDpsFI9Wbharwcysa83fxb402/wDH9oP/AIKpv/kmoofFfjIZxfaF+OlTf/JNDpu4KR6TZ2X9n2LQ+fLOWkklLyuWOXYsQMk4UZwB2AAqe2+6a8ym8WeM9nN9oX4aVN/8k0kPivxmE4vtC/HSpv8A5Jp+zfNuHMrHp10fkAp8P+rFeWz+LPGZxuvtC/DSpv8A5Jp6+LPGYQYvtB/8FU3/AMk0lTd9w5tD0O+tknuoJGeVTESVCSsoORj5gDhvbPQ803+xoPs3leffbfIWDP22XdtBznO7O71b7x9a85k8V+MjMM32hZ/7BU3/AMk1OPFvjTH/AB/aD/4Kpv8A5JpqDT3E2juzp0TagZDLc5MwmwLmTGQu3GM424529CecZq/b2yWdqIo3ldQScyytI3JJ+8xJ7/gOK8uTxX4yM5P27Qs/9gqb/wCSamfxb402832g/wDgqm/+SaSg7bjuj0e25kNTTnEZryyDxX4yBO2+0L8dKm/+Sakm8WeM/LOb7Qse2lTf/JNJU3yj5tT0y1+5SXZ4FeZweLPGYX5b7Qvx0qb/AOSabceK/GRxuvtC/DSpv/kmj2b5Q5lc9IsbLyb65vDPK5uFRfLZyUQKD91c4BOTkjGePSnXB/fivOI/FnjMRjF9oP8A4Kpv/kmoZfFfjIyjN9oWf+wVN/8AJNDpu24ubU9YU/KtUx/x9GvOx4s8abf+P7Qf/BVN/wDJNVh4q8Y+dn7doef+wXN/8k0SpvTUFJHrEn+raqtt/rDXnMni3xpsOb7Qf/BVN/8AJNQQeK/GQc7b7Q/x0ub/AOSaJU3fcakep3R/dGmWh4rzO48WeMzH819oWPbSpv8A5JpsHivxkB8t9oX46VN/8k0ezfNuLm0PTLz7wqW2+4K8tufFfjI43X2hfhpU3/yTT4fFnjMKMX2hfjpU3/yTR7N824+ZWPRZ/wDj4/Gran91XlE3irxiZcm90PPtpc3/AMk1YXxZ4z8v/j+0Lp/0Cpv/AJJpRpu+4OR6Ep/0g/WnajYQ39vGtz5hWN94VJWRWOCMMARuXnocivMh4q8Yed/x+6Hn/sFzf/JFWH8WeMzHzfaFj/sFTf8AyTSjTdnqNyRtyeF7mAWdroFzJZWVt1iF1IpbMiE888BFYAe4HArrrr/V15dF4q8YBuL3Q/x0ub/5IqSfxX4yKfNfaH+Glzf/ACTRyScbNiur3PSrU8U27+8K81g8VeMR9290P8dLm/8Akmm3Pirxicbr3Q/w0ub/AOSKXs3y7lcyueoQH5PwqtNxPXncPivxiFGL7Q/x0ub/AOSahm8U+MDJk3uh59tLm/8AkilKk7bi5lc9UX/V1TTiY/WvPU8V+Mdv/H9of/grm/8Akmq//CU+LxL/AMfuiZ/7Bc3/AMkUOk+41I9Tl/1RqvAfmNedt4r8Y+Wc32h/+Cub/wCSagj8U+L1bi90T/wWS/8AyRSdJ33DmVj065+5TbdvlrzebxV4wZOb3Q/w0ub/AOSKZD4p8YDpe6J+Olzf/JFHsnfcOZWPSbn7wNSRHKivMp/FPjA4ze6J+Glzf/JFOi8VeMABi90P8dLm/wDkij2TvuHMj0N+JuasD/V/hXmEvijxfv5vdE/DTJf/AJIqRPFPjDb/AMfuh/8Agrm/+SKSpPuPmR6EhxKanl5jrzA+KPF3mf8AH7omf+wZL/8AJFSnxT4wKc3uh/8Agrm/+SKFSfcHJHocPOQeaV4o4YVjiRY0QYVVGAB6AV5snijxcH4vdF/8Fkv/AMkVJJ4o8Xleb3RP/BXN/wDJFJUnbcOZXPRoDxTJ/vZrzmLxR4v7Xuifjpkv/wAkUTeKPFx5N7on4aZL/wDJFL2TtuHMj0qLlahfiWvPIvFPi/bxe6J/4LJf/kimSeJ/F27JvdF/8Fkv/wAkUOk7bhzI9NHMdQKcSGvPU8U+L9v/AB+6J/4LJv8A5IqE+J/Fof8A4/dF/wDBZL/8kUOk+4+ZHpsnMdRQnDV57/wlPi8pj7bon/gsl/8AkioV8T+Lg3F7ov8A4LJf/kih0nfcXMj0qcd6ITwK85fxR4vK83uif+CyX/5IpkfijxcDxe6L/wCCyX/5Io9i77j50ejT8NmpYzla81l8T+Lj1vdF/wDBZL/8kUR+KPF3a90X/wAFkv8A8kUexd9w51Y9Db5Zag1DR7PVVja7jYvGrKro7IwVhhlyCDg4GR7D0rgZPE3i3dk3ui/+CyX/AOSKenijxfj/AI/dE/8ABZL/APJFCpST3DmR39vHHbRpDCixxxqFRFGAoHAAqxIMx15ofE/i0P8A8fmi/wDgsl/+SKl/4SjxeV/4/dE/8Fkv/wAkUlRfVi5ktj0KJsGnTjgGvNh4n8WhuL3Rf/BZL/8AJFWIfEviuS4iSW80YozgMF02UHGecH7Qf5UvYu24+ZXPQFb5aKjBwKKyLP/Z)

#### **1.2.2. Doğrusal Programlamada Çözüm Yöntemleri**

Doğrusal programlama modelleri aşağıda sıralanan yöntemler ile çözülebilir:

1. Grafik Yöntemi

2. Cebirsel Yöntem

3. Simpleks Yöntem

4. İleri Doğrusal Programlama Çözüm Yöntemleri

Optimizasyon problemlerinin optimum değerlerini bulmak için bu yöntemlerden sadece grafik yöntemi kullanılacaktır.

##### **1.2.2.1. Grafik Yöntemi ve Örnekler**

Bir doğrusal programlama probleminin grafik çözümünde aşağıdaki adımlar izlenir:

1. Değişkenlerin koordinat sisteminin yatay ve dikey eksenlerine yerleştirilmesi,

2. Kısıtlayıcı fonksiyonların grafiğinin çizilmesi,

3. Uygun çözüm bölgesinin belirlenmesi,

4. En iyi çözümün araştırılması.

Soru 1:

Bir imalatçı A ve B olmak üzere iki farklı modelde karton katlama makinası satın almak istiyor. Model-A’daki makine dakikada 30 karton katlıyor ve 1 kişinin yardımına ihtiyacı var. Model-B ise dakikada 50 karton katlıyor ve 2 kişinin yardımına ihtiyaç duyuyor. İmalatçı dakikada 320 karton katlamak istiyor ve 12 kişiden fazlası kullanılamayacaktır diyor. Model-A’nın maliyeti 150 birim, Model-B’nin maliyeti ise 200 birimdir. İmalatçı maliyeti en küçük olacak biçimde her tip makinadan kaç tane alması modelini matematiksel olarak çözünüz.

: A katlama makinasından kaç tane alınması gerektiği

: B katlama makinasından kaç tane alınması gerektiği

Soruda Model-A’nın maliyeti 150 birim, Model-B’nin maliyeti ise 200 birim olduğundan ve imalatçı da maliyeti en küçük olacak biçimde talep ettiğinden amaç fonksiyonumuz:

Sorudan hareketle Model-A’daki makine dakikada 30 karton katlıyor ve 1 kişinin yardımına ihtiyacı var. Model-B ise dakikada 50 karton katlıyor ve 2 kişinin yardımına ihtiyaç duyuyor. İmalatçı dakikada 320 karton katlamak istediğinden ve 12 kişiden fazlası kullanılamayacağından karton kısıtı(1) ve kişi kısıtı(2) olacak şekilde teknolojik(sistem) kısıtlarımız aşağıdaki gibi olacaktır:

(1).

(2).

İfadelerimiz sıfırdan büyük olacağından -yani üretmemiz gereken miktar negatif olamayacağından- negatif olmama kısıtımız da aşağıdaki gibi yazılmalıdır:

Bu sistem kısıtlarımızın grafiğini çizelim.

**Grafik 1.1: Soru 1’deki Sistem Kısıtlarının Koordinat Sistemindeki Görünümü**

Sistem kısıtlarımız koordinat düzleminde Grafik 1.1’deki gibi görünecektir. (1) kısıtımız “”, (2) kısıtımız ise “” ibaresi olduğundan “” kısıtı yukarı taraf, ” kısıtı ise aşağı taraf taranmalıdır. Bu açıklamaya göre uygun çözüm alanımız ve köşe noktalarımız ise aşağıdaki gibi olacaktır:

**Grafik 1.2:Soru 1’in Uygun Çözüm Alanı ve Köşe Noktaları**

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu noktalarımızı amaç fonksiyonunda yerine koyacak olursak sonuçlar aşağıdaki gibi olacaktır:

**Tablo 1.1: Grafik 1.2’deki Köşe Noktalarının Amaç Fonksiyonundaki Değer Tablosu**

|  |  |
| --- | --- |
| Noktalar |  |
| (10.6,0) | 1590 |
| (12,0) | 1800 |
| **(4,4)** | **1400** |

Tablo 1.1’de görüldüğü üzere noktası bizim optimal çözümümüzdür. Yani imalatçı, A modelinden ve B modelinden 4’er tane karton katlama makinesi satın alırsa, kısıtları sağlayacak şekilde minimum maliyette(1400) en iyi sonucu elde etmiş olur.

Soru 2:

Bir şirket 2 çeşit ayakkabı üretmektedir. Her biri için gereken iş gücü sırasıyla 2 ve 3 saattir ve gereken malzeme miktarı sırasıyla 3 ve 2 kg/çifttir. Şirketin aylık 30 saat iş gücü ve 40kg malzemesi bulunmaktadır. Ayakkabıların çiftlerinden sırasıyla 4 ve 6 birim kar elde edilmektedir. Şirket kârı en yüksek olacak şekilde her bir çeşit ayakkabıyı ne kadar üretmesi modelini matematiksel olarak çözünüz.

x1: 1. ayakkabı x2: 2. ayakkabı

Soru-2’de 1. ayakkabının maliyeti 4 birim, 2. ayakkabının maliyeti ise 6 birim olduğundan ve şirket kârı en yüksek olacak biçimde ayakkabı üretmek istediğinden amaç fonksiyonumuz:

Sorudan hareketle her bir ayakkabı için gereken iş gücü sırasıyla 2 ve 3 saattir ve gereken malzeme miktarı sırasıyla 3 ve 2 kg/çifttir. Şirketin aylık 30 saat iş gücü ve 40kg malzemesi bulunduğundan saat kısıtı(1) ve malzeme miktarı(2) olacak şekilde sistem kısıtlarımız aşağıdaki gibi olacaktır:

(1).

(2).

Tabi ki ifadelerimiz sıfırdan büyük olacağından -yani üretmemiz gereken miktar negatif olamayacağından- negatif olmama kısıtımız da aşağıdaki gibi yazılmalıdır:

Sistem kısıtlarımızın grafiğini çizecek olursak:

**Grafik 1.3: Soru 2’deki Sistem Kısıtlarının Koordinat Sistemindeki Görünümü**

Sistem kısıtlarımız koordinat düzleminde Grafik 1.3’teki gibi görünecektir. Her iki kısıtta da “” ibaresi olduğundan aşağı taraf taranmalıdır. Bu açıklamaya göre uygun çözüm alanı ve köşe noktaları ise aşağıdaki gibi olacaktır:

**Grafik 1.4: Soru 2’nin Uygun Çözüm Alanı ve Köşe Noktaları**

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu noktalarımızı amaç fonksiyonunda yerine koyacak olursak sonuçlar aşağıdaki gibi olacaktır:

**Tablo 1.2: Grafik 1.4’teki Köşe Noktalarının Amaç Fonksiyonundaki Değer Tablosu**

|  |  |
| --- | --- |
| Noktalar |  |
| (0,0) | 0 |
| **(0,10)** | **60** |
| (13.3,0) | 53,2 |
| **(12,2)** | **60** |

Tablo 1.2’de görüldüğü üzere iki farklı optimal çözüm elde edildi. Yani şirket 1. ayakkabıdan 0, 2. ayakkabıdan 10 adet üretse; veyahut 1. ayakkabıdan 12 adet, 2. ayakkabıdan ise 2 adet ayakkabı üretse dahi maksimum kârı(60) elde etmiş olur.

Soru 3:

Giapetto tahtadan oyuncak asker ve tren yapmaktadır. Satış fiyatları, bir oyuncak asker için $3, bir oyuncak tren için $2'dır. Her bir asker için 2 saat montaj ve 1 saat marangozluk gerekirken, her bir tren için 1 saat montaj ve 1 saat marangozluk gerekmektedir. Eldeki hammadde miktarı sınırsızdır, fakat haftada en çok 100 saat montaj ve 80 saat marangozluk kullanabilen Giapetto'nun haftada en fazla 40 oyuncak asker satabileceğini göz önünde bulundurarak kârını maksimum yapmak için hangi oyuncaktan haftada kaç adet üretmesi gerektiğini bulunuz.

= Bir haftada üretilen asker sayısı = Bir haftada üretilen tren sayısı

Soru-3’te bir oyuncak askerin maliyeti 3 birim, oyuncak trenin maliyeti ise 2 birim olduğundan ve Giapetto kârı en yüksek yapmak istediğinden amaç fonksiyonumuz:

Sorudan hareketle her bir asker için 2 saat montaj ve 1 saat marangozluk gerekirken, her bir tren için 1 saat montaj ve 1 saat marangozluk gerekmektedir. Eldeki hammadde miktarı sınırsızdır, fakat haftada en çok 100 saat montaj ve 80 saat marangozluk kullanabilen Giapetto'nun haftada en fazla 40 oyuncak asker satabileceğini göz önünde bulundurarak montaj kısıtı(1), marangozluk kısıtı(2) ve talep kısıtı(3) olacak şekilde sistem kısıtlarımız aşağıdaki gibi olmalıdır:

(1).

(2).

(3).

Negatif olmama kısıtımız ise:

Sistem kısıtlarımızın grafiğini çizecek olursak:

**Grafik 1.5**: **Soru 3’teki Sistem Kısıtlarının Koordinat Sistemindeki Görünümü**

Sistem kısıtlarımız koordinat düzleminde Grafik 1.5’teki gibi görünecektir. Her iki kısıtta da “” ibaresi olduğundan aşağı taraf taranmalıdır. Bu açıklamaya göre uygun çözüm alanı ve köşe noktaları ise aşağıdaki gibidir:

**Grafik 1.6: Soru 3’ün Uygun Çözüm Alanı ve Köşe Noktaları**

metin, harita içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu noktalarımızı amaç fonksiyonunda yerine koyacak olursak:

**Tablo 1.3: Grafik 2.6’teki Köşe Noktalarının Amaç Fonksiyonundaki Değer Tablosu**

|  |  |
| --- | --- |
| Noktalar |  |
| (0,0) | 0 |
| (0,80) | 160 |
| **(20,60)** | **180** |
| (40,20) | 160 |
| (40,0) | 120 |

Tablo 1.3’te görüldüğü gibi noktası bizim optimal çözümümüz olacaktır. Yani Giapetto haftada 20 adet oyuncak asker, 60 adet de oyuncak tren üretirse, kısıtları sağlayacak şekilde maksimum kârda(180) en iyi çözümü elde etmiş olur.

Soru 4:

Bir elbise dikim firması, fabrika işçileri için iş elbisesi(üniforma) siparişi almaktadır ve sipariş için 240m kumaşı mevcuttur. Üniformalar A ve B tipli model olarak hazırlanacaktır. Her bir A tipli model yaklaşık 3 saat, B tipli model ise 1 saat işlem gerektirmekte ve bu üretim için günlük toplam 320 saatlik bir işgücü mevcut bulunmaktadır. A ve B tip her bir modelin gerektirdiği kumaş miktarları ise sırası ile 2m ve 1m kadardır. A ve B tip bir her bir modelin satışından elde edilecek kar ise sırasıyla 25 ve 20 TL’dir. Günlük üretimden elde edilecek olan karın maksimum olması için hangi modelden ne kadar üretilmelidir?

Soru-4’te A tip elbise modelinin maliyeti 25 birim, B tip elbise modelinin maliyeti ise 20 birim olduğundan ve dikim firması kârı en yüksek yapmak istediğinden amaç fonksiyonumuz aşağıdaki gibi olacaktır:

Sorudan hareketle sipariş için 240m kumaşı mevcuttur. Üniformalar A ve B tipli model olarak hazırlanacaktır. Her bir A tipli model yaklaşık 3 saat, B tipli model ise 1 saat işlem gerektirmekte ve bu üretim için günlük toplam 320 saatlik bir işgücü mevcut bulunmaktadır. A ve B tip her bir modelin gerektirdiği kumaş miktarları ise sırası ile 2m ve 1m kadardır. Bu istekler doğrultusunda saat(1) ve işgücü(2) olacak şekilde kısıtlarımız şu şekildedir:

(1).

(2).

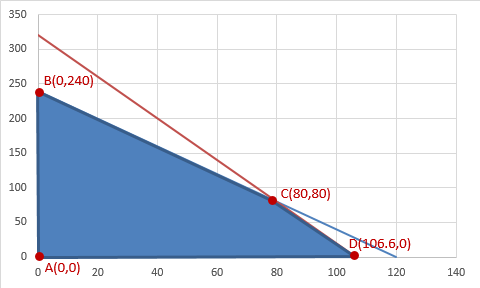
Negatif olmama kısıtımız ise:

Sistem kısıtlarımızın grafiğini çizecek olursak:

**Grafik 1.7:** **Soru 4’teki Sistem Kısıtlarının Koordinat Sistemindeki Görünümü**

Sistem kısıtlarımız koordinat düzleminde Grafik 1.7’deki gibi görünecektir. Her iki kısıtta da “” ibaresi olduğundan aşağı tarafa taranmalıdır. Bu açıklamaya göre uygun çözüm alanı ve köşe noktaları ise aşağıdaki gibi olacaktır:

**Grafik 1.8:** **Soru 4’ün Uygun Çözüm Alanı ve Köşe Noktaları**

****

Bu noktalarımızı amaç fonksiyonunda yerine koyacak olursak:

**Tablo 1.4: Grafik 1.8’deki Köşe Noktalarının Amaç Fonksiyonundaki Değer Tablosu**

|  |  |
| --- | --- |
| Noktalar |  |
| (0,0) | 0 |
| **(0,240)** | **4800** |
| (80,80) | 3600 |
| (106.6,0) | 2665 |

Tablo 1.4’te görüldüğü gibi noktası bizim optimal çözümümüz olacaktır. Yani firma sadece B tip elbise modelinde 240 adet üretirse kısıtları sağlayacak şekilde maksimum kârı elde etmiş olur.

Soru-5:

Bu soru da ise optimal sonucu olmayan bir örnek ele alınmaktadır. Aşağıdaki doğrusal programlama problemini grafik yöntemiyle çözünüz.

Bu sistem kısıtlarını ele aldığımızda grafiğimiz aşağıdaki gibi olacaktır:

**Grafik 1.9: Soru 5’teki Sistem Kısıtlarının Koordinat Sistemindeki Görünümü**

Uygun çözüm alanımız ise:

**Grafik 1.10: Soru 5’nin Uygun Çözüm Alanı**

bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu

Grafik 1.10’da görüldüğü gibi ortak çözüm alanı olmadığından bu sorunun uygun çözümleri ve optimal çözümü yoktur.

# İKİNCİ BÖLÜM

## 2. MiniZinc’e İLE OPTİMİZASYON

### **MiniZinc Nedir ?**

MiniZinc, tamsayılar ve gerçek sayılar üzerinde kısıtlı optimizasyon ve karar problemlerini belirlemek için kullanılan bir dildir. Bir MiniZinc modeli sorunun nasıl çözüleceğini  diretmez .MiniZinc derleyici, onu Kısıt Programlama (CP), Karışık Tamsayılı Doğrusal Programlama (MIP) veya Mantıksal İfadenin Sağlanabilirliği (SAT) gibi çok çeşitli çözücüler için uygun farklı formlara dönüştürebilir.

MiniZinc dili, dizin kümeleri üzerinden toplamlar veya çıkarımlar, if-then-else ifadeleri ve mantıksal bağlaçlar gibi bilinen gösterimleri kullanarak modellerin, problemin matematiksel bir formülasyonuna yakın bir şekilde yazmasına izin verir. Ayrıca MiniZinc, kullanıcıların modellerini yapılandırmasına izin veren tahminleri ve işlevleri tanımlamayı destekler (normal programlama dillerindeki prosedürlere ve işlevlere benzer).

MiniZinc modelleri genellikle parametriktir yani bir problem örneğini ele almaktan ziyade  problemin tüm sınıfını tanımlar. Böylelikle bir araç yönlendirme problemi modeli, gelecek hafta için güncellenen müşteri talepleriyle somutlaştırılarak haftalık planlar oluşturmak için yeniden kullanılabilir.

MiniZinc, farklı arka plan derleyicilerine kolayca arayüz oluşturacak şekilde tasarlanmıştır. Bunu, MiniZinc modelini ve veri dosyasını FlatZinc modeline dönüştürerek yapar. Problem optimizasyon problemi ise  FlatZinc modelleri nesnel fonksiyon tanımın yanı sıra değişken ifadelerinden ve kısıt tanımlarından oluşur. Ayrıca modele ait açıklamalar veya notlar modelin sunduğu anlamından bağımsız, kullanıcıya derleyicinin davranışına ince ayar yapmasına izin verir.

##### **2.1.1. MinZinc’in Özellikleri**

MiniZinc yüksek seviyeli, çoğunlukla birinci dereceden işlevsel bir modelleme dilidir. MiniZinc:

* Matematiksel gösterime benzer söz dizimi  (automatic coercions, overloading, iteration, sets, arrays)
* İfade kısıtlamaları (sonlu alan, küme, lineer aritmetik, tamsayı)
* Çeşitli sorunlara destek
* Verilerin modelden ayrılması
* Yüksek seviyeli veri yapıları (kümeler, diziler, numaralandırılmış türler)
* Genişletilebilirlik (kullanıcı tanımlı işlevler ve tahminler)
* Güvenilirlik (tip kontrolü, örnekleme kontrolü, iddialar)
* Çözücüden bağımsız modelleme gibi özelliklere sahiptir

##### **2.1.2. MinZinc’te Kullanılan Bazı Komutlar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Değişken Türleri** | **Açıklama** | **Genel Kullanım** | **Örnek** |
| **int** | Diğer dillerde olduğu gibi tam sayı tanımlamak için kullanılabilir. | int: <var-name> | int: |
| **var** | Belirli sayılar aralığında değişken tanımlanmak için kullanılır. | var <i>..<u>: <var-name> | var 1..100: |
| **var int** | Belirli sayılar arasında olmayan değişken tanımlanabilir. | var int: <var-name> | var int: |
| **constraint** | Modelimizdeki kısıtlarımızı tanımlamak için kullanılan değişken türüdür. | constraint | constraint |
| **solve minimize** | Minimum maliyet istendiğinde kullanılan değişken tipidir. | solve minimize | solve minimize |
| **solve maximize** | Maximum kar istendiğinde kullanılan değişken tipidir. | solve maximize | solve maximize |
| **output** | Uygun çözümleri ekrana çıktı vermek için kullanılır. | output [<açıklama>\(parametre)\”,  .  .  .]; | output[*"A makinası = \(x1)\n"*,  *"B makinası = \(x2)\n"*]; |

##### **2.1.3. Bir Optimizasyon Modeline Genel Bakış**

Kavramsal olarak, bir MiniZinc problemi iki bölümden oluşur:

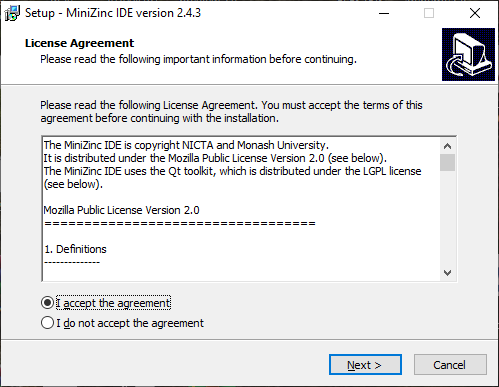
1.Model: Problemin ana hatlarını tanımlar.

2.Veri: Model için girdileri tanımlar.

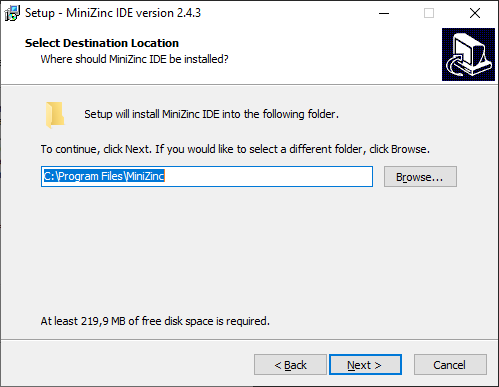
İki geniş problem sınıfı vardır: Uygunluk ve optimizasyon. Uygunluk problemlerinde tüm çözümler eşit derecede iyi kabul edilirken, optimizasyon problemlerinde çözümler amaca göre sıralanır. Amaç optimal çözümü bulmaktır.

## 2.2. Kurulum

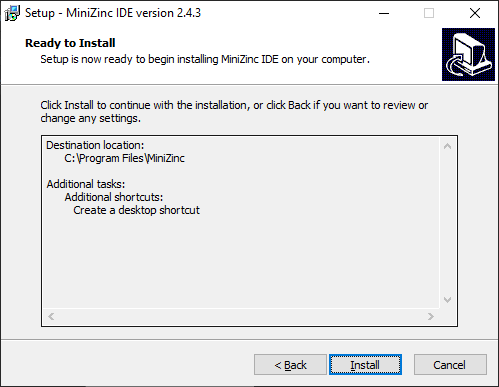
Bu bölümde Windows işletim sistemime MiniZinc programının kurulumu ele alınacaktır. MiniZinc IDE’yi <https://www.minizinc.org/software.html> adresinden indirilebilir. Adresini açtığımızda birçok işletim sistemine ait yükleme paketleri mevcuttur. 32 bit Windows yüklemeleri artık desteklenmemektedir bu sebeple 64bit mimariye sahip olan yükleme paketi indirilmelidir. Windows yükleyici paketini indirdikten ve kurmak için yükleme paketine çift tıkladıktan sonra ilk adım olan aşağıdaki pencere ile karşılaşacağız.



Karşımıza birçok yükleyicide olduğu gibi bazı kabul edilmesi gereken şartlar ve sözleşmeler mevcut bunları kabul etmek adına “I agree the agreement” seçeneğine tıkladıktan sonra “Next” ile bir sonraki adıma geçelim.



Burada ise varsayılan olarak MiniZinc programının kurulacağı dizini seçtikten sonra Next tuşuna tıklayarak aşağıdaki son adıma geçelim.



Görüldüğü gibi son adımda ise yükleme işlemi bittikten sonra kurulumu tamamlanmaktadır. Kurulumu tamamladıktan sonra .mzn, .dzn ve .fzn dosya uzantılarına sahip herhangi bir MiniZinc dosyasını çift tıklayarak IDE’de açabilirsiniz.

## 2.3. MiniZinc ile İlk Adımlar

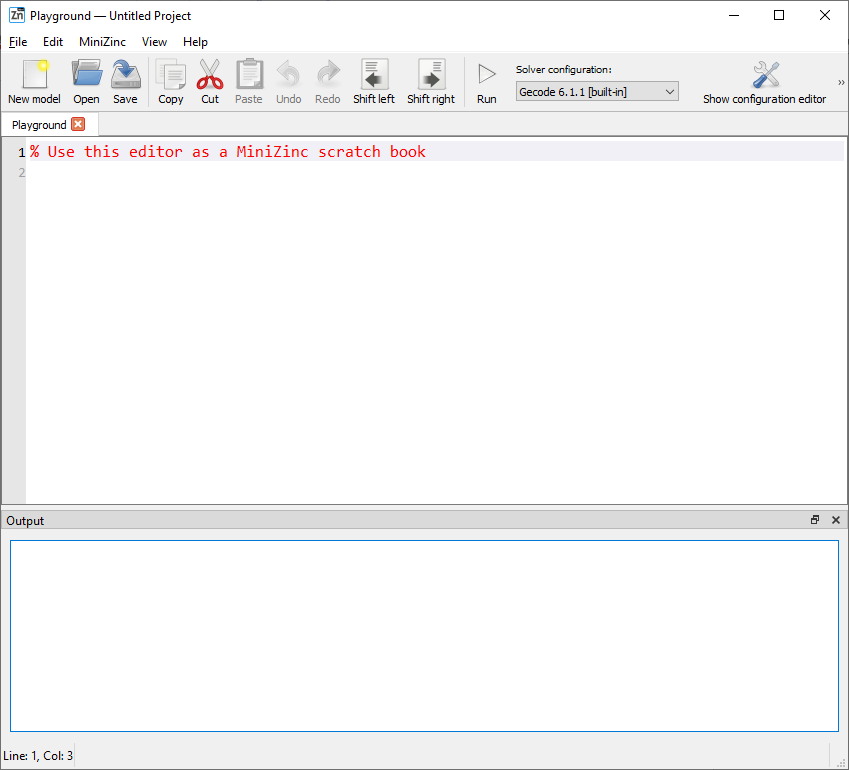
MiniZinc IDE, MiniZinc derleyicisini ve önceden yapılandırılmış birkaç çözücüyü içerir, böylece hemen başlayabilirsiniz. Bu bölümde, bazı çok temel örnekleri kullanarak MiniZinc IDE ve tanıtılmaktadır.

### **2.3.1. MiniZinc IDE**

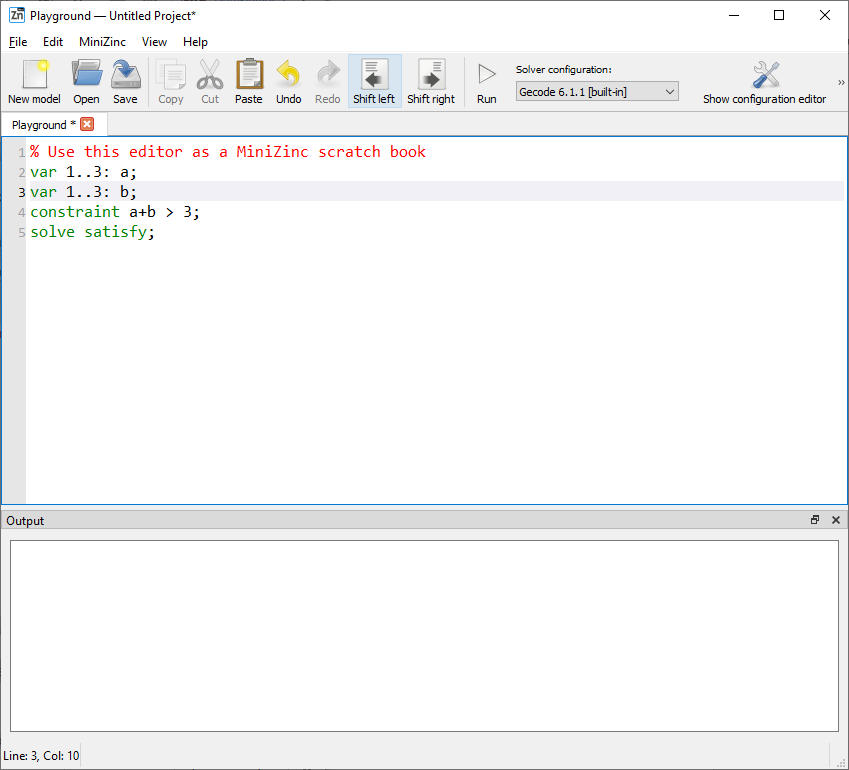
MiniZinc IDE, MiniZinc'in işlevselliğinin çoğuna basit bir arayüz sağlar. Model ve veri dosyalarını düzenlemenizi, MiniZinc tarafından desteklenen çözücülerden herhangi birini çözmenizi, hata ayıklama ve profil oluşturma araçlarını çalıştırmanızı ve çevrimiçi kurslara çözümler sunmanızı sağlar.

MiniZinc IDE'yi ilk kez açtığınızda, bir güncelleme olduğunda size bildirilmek isteyip istemediğinizi soracaktır. IDE'yi kaynaklardan yüklediyseniz, MiniZinc derleyicisinin kurulumunu bulmanızı isteyebilir.

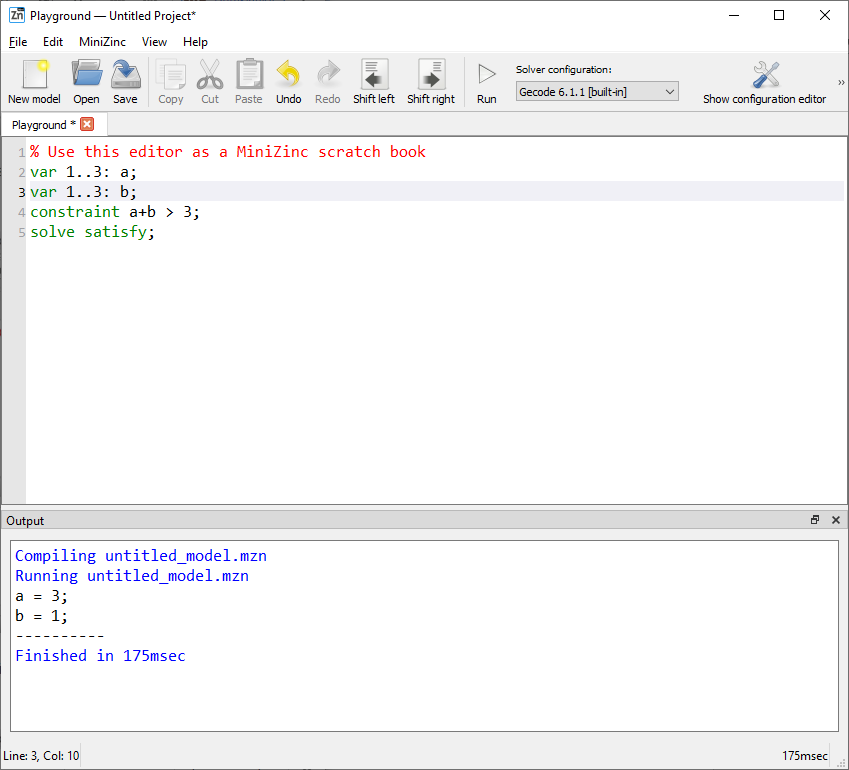
IDE sizi aşağıdaki gibi görünen bir pencere olan MiniZinc “PlayGround” ile karşılayacaktır.



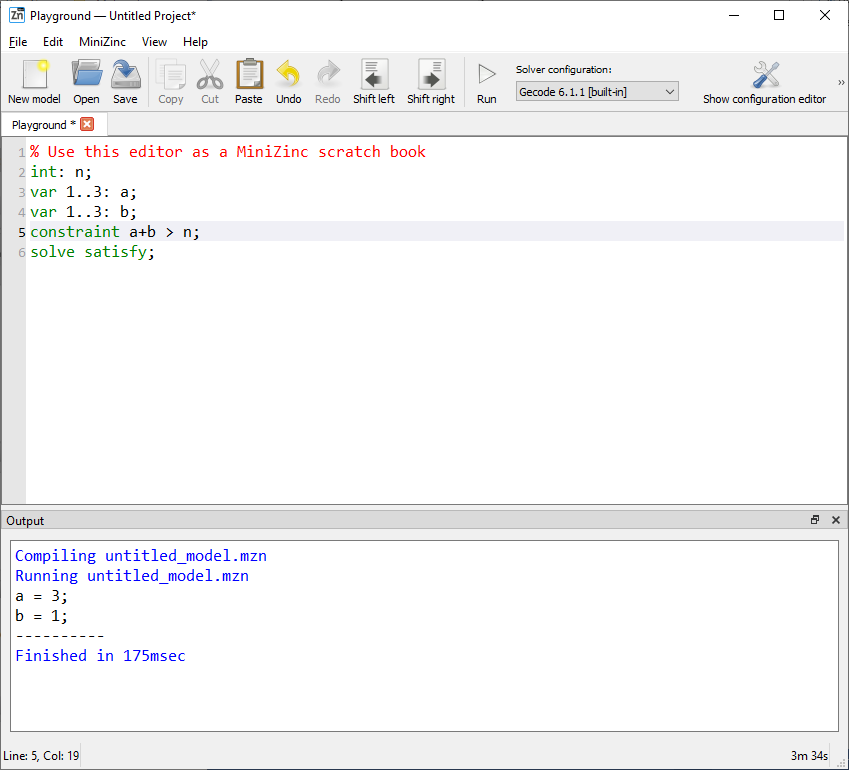
İlk MiniZinc modelinizi yazmaya başlayabilirsiniz! Çok basit bir şey deneyelim:



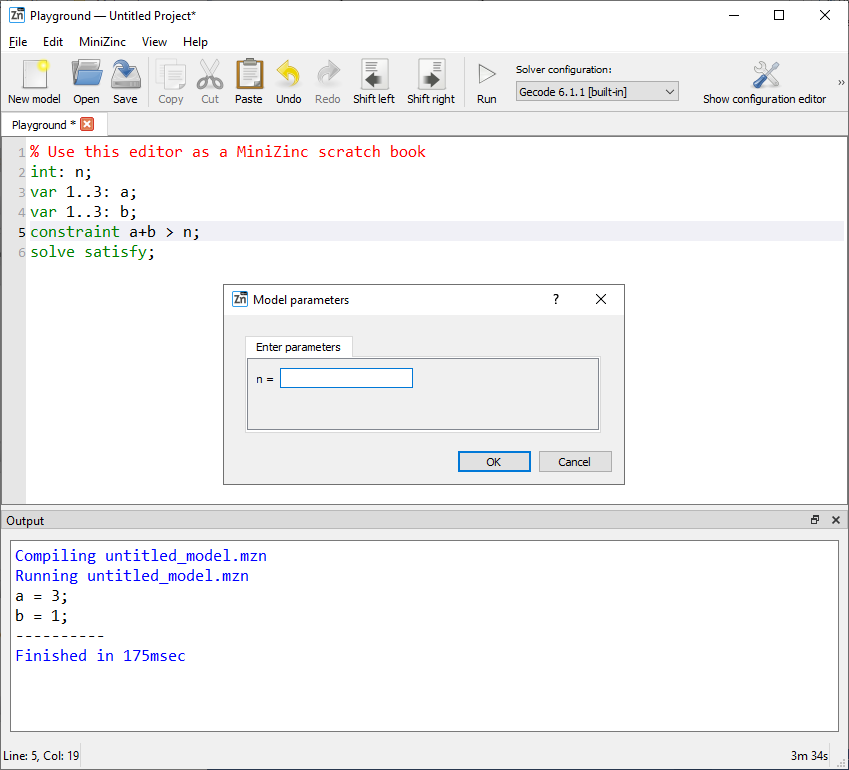
Modeli çözmek için araç çubuğundaki Çalıştır düğmesine tıklayın veya Ctrl + R klavye kısa yolunu kullanın (veya macOS'ta + R komutu ):



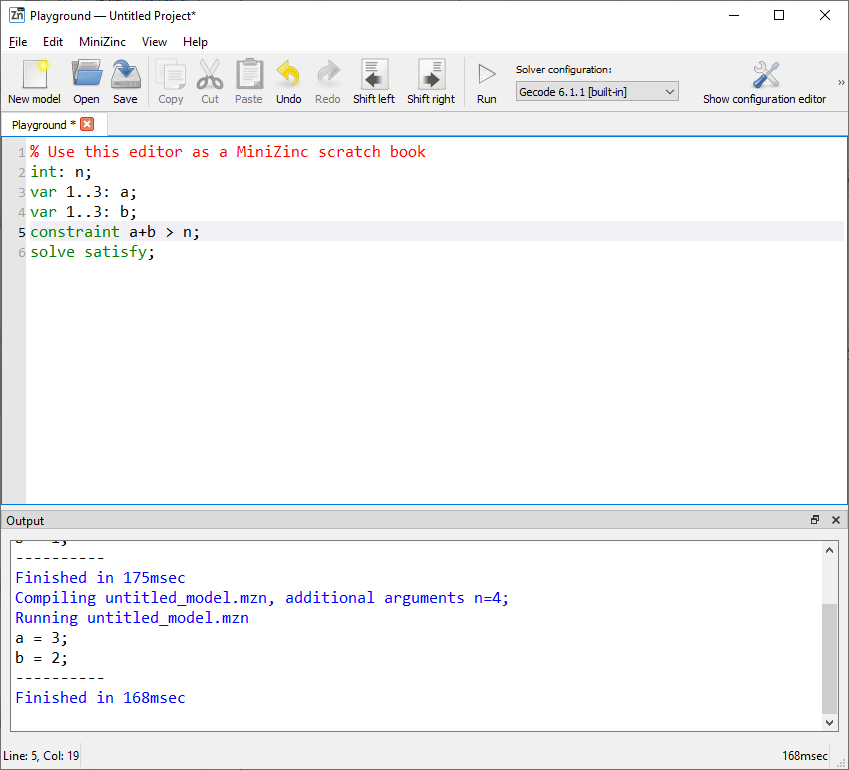
Gördüğünüz gibi, girdiğiniz soruna bir çözüm görüntüleyen bir çıkış penceresi açılır. Şimdi ek veri gerektiren bir model deneyelim.



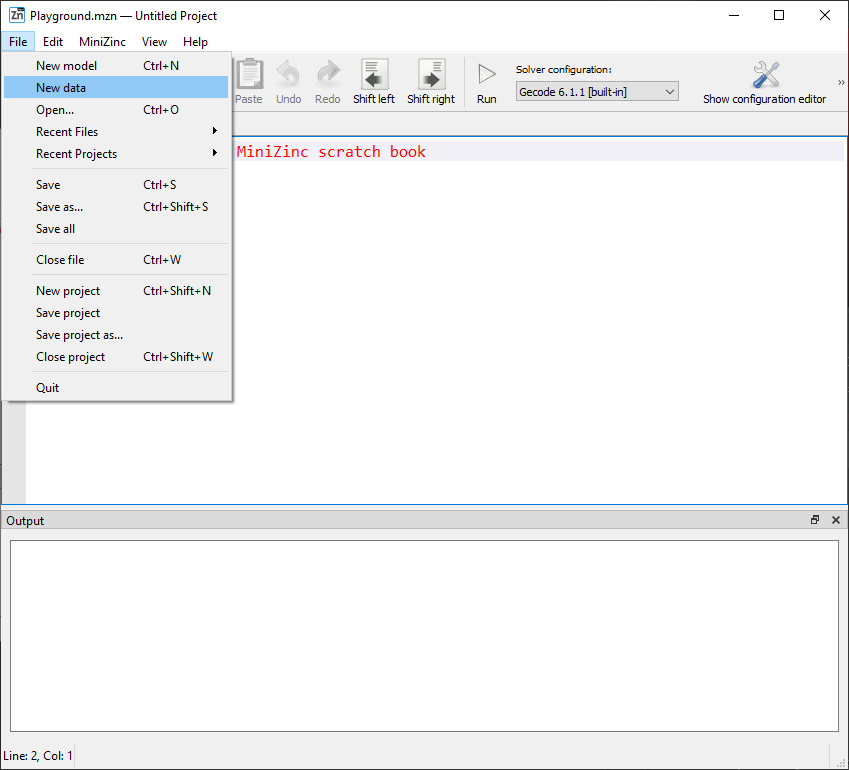
Bu modeli çalıştırdığınızda, IDE sizden *n* parametresi için bir değer girmenizi ister.



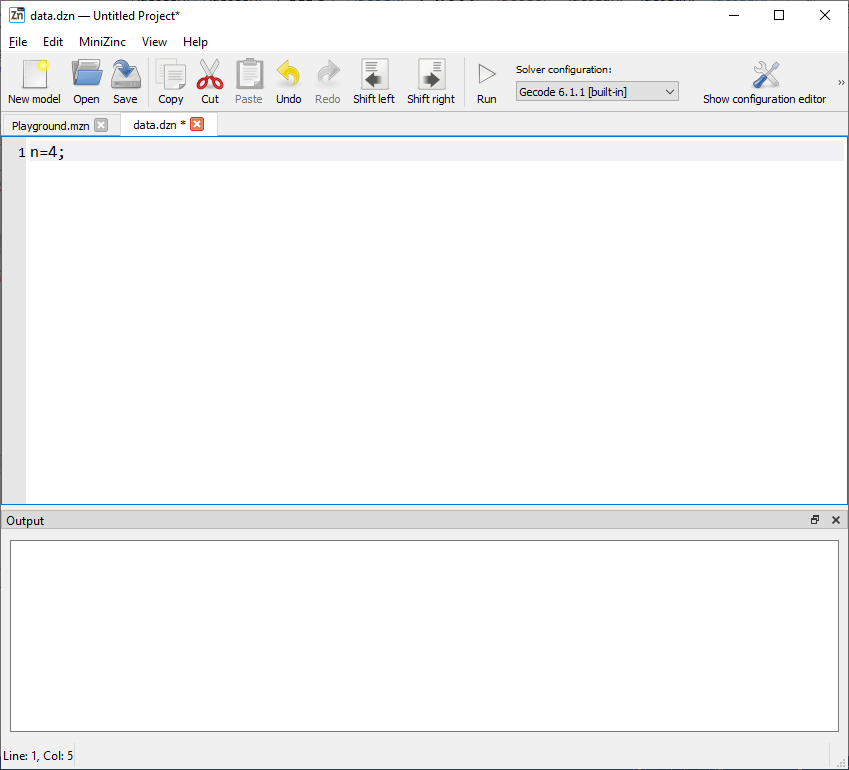
Örneğin n için 4 değerini girelim ve modelin çözümünü görmek için çalıştır düğmesine tıkladığımızda aşağıdaki gibi sonuçlar elde edilecektir.



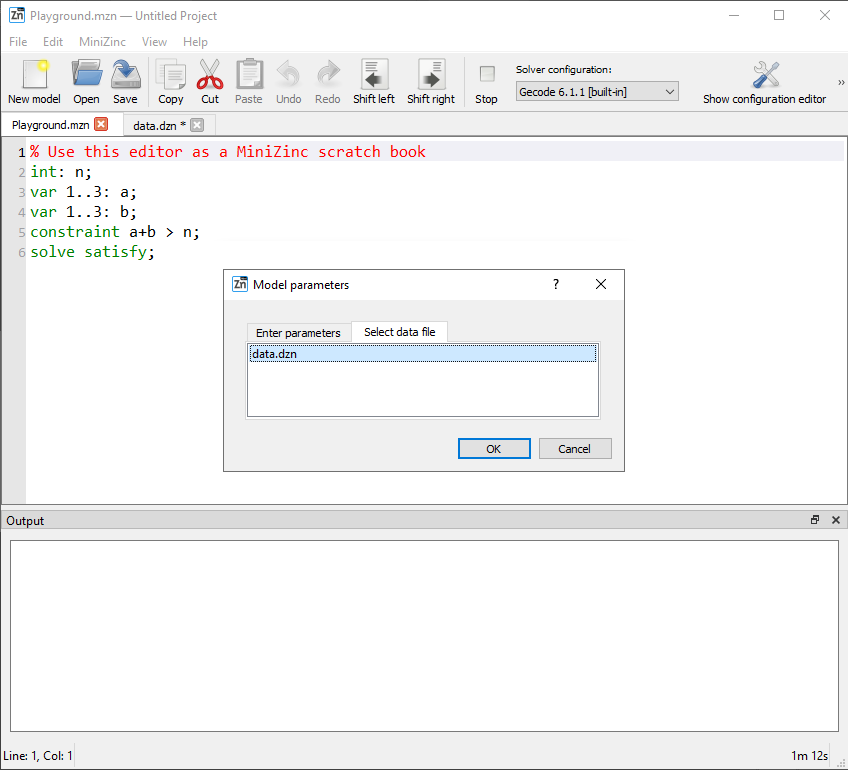
Ayrıca n için verimizi bir dosyadan aktarabiliriz. Bunun için aşağıda ki resim de görüldüğü gibi File-New data ile veri dosyamızı oluşturalım.



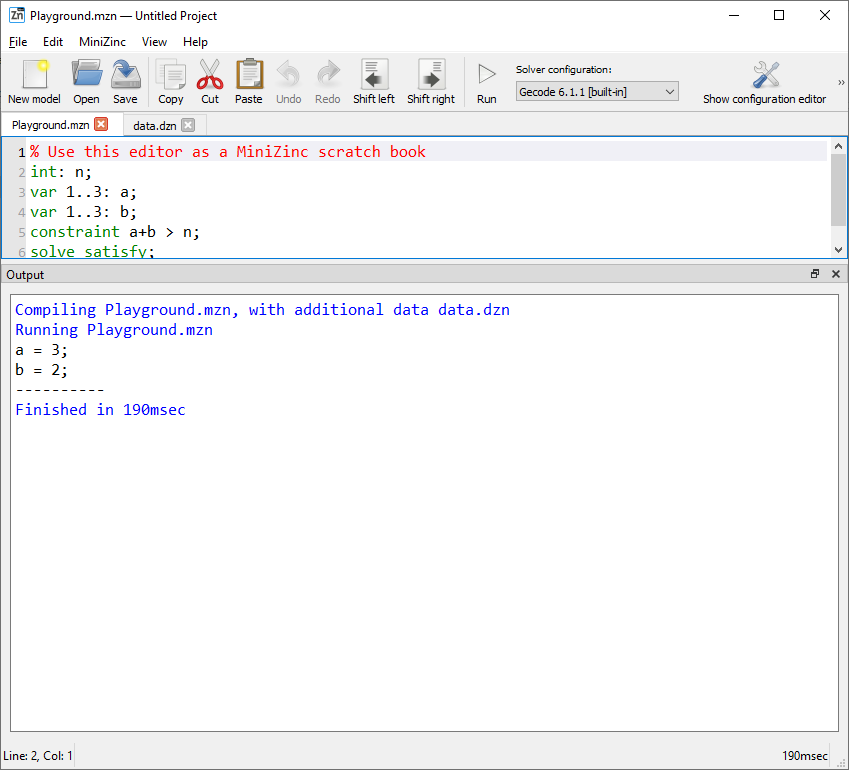
Oluşturduktan sonra n değerini 4 verelim.



Son olarak n değerimizi girdikten sonra Playground sekmesinde “Run” tuşuna tıkladığımızda, IDE size veri seçme seçeneği sunar.



Bu pencerede verimizi barındıran “data” adlı dosyayı seçelim ve modelimizin a ve b’nin sonuçlarını görmek için “OK” tuşuna tıklayalım.



Görüldüğü gibi a için 3 ve b için 2 değerleri elde edildi. Elbette oluşturduğunuz modelinizi bir dosyaya kaydedebilir ve bir dosyadan yükleyebilirsiniz.

# ÜÇÜNCÜ BÖLÜM

## 3. MiniZinc İLE UYGULAMALAR

### **3.1. Optimizasyon Problemlerinin Minizinc’e Aktarılması ve Çözümü**

Bu bölümde daha önce 4 adet ele aldığımız optimizasyon problemlerini kullanarak bu problemlerin MiniZinc’e nasıl aktarıldığı ve modelin temel yapısı anlatılmaktadır.

Soru 1’in MiniZinc ile Çözümü:

İlk örneğimizden hareketle iki farklı tipte karton katlama makinası mevcuttur. x1, A katlama makinasından kaç tane alınması gerektiğini; x2 ise B katlama makinasından kaç tane alınması gerektiğini temsil etmektedir. Bu değişkenleri MiniZinc’te tanımlayalım:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-1’deki teknolojik(sistem) kısıtlar:

Bu teknolojik kısıtları ekledikten sonra MiniZinc’teki görünüm aşağıdaki gibi olacaktır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Görüldüğü gibi kısıtlarımızı “constraint” komutu ile tanımlayabiliriz.

İfadelerimiz sıfırdan büyük olacağından -yani üretmemiz gereken miktar negatif olamayacağından- negatif olmama kısıtımız aşağıdaki gibi yazılmalıdır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Söz konusu Soru-1’deki amaç fonksiyonumuz olarak tanımlanmıştı. Bu amaç fonksiyonumuzu MiniZinc’e aktaralım.

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Çıktılarımızı yazdırabilmek için ise aşağıdaki komut eklenmelidir:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Uygun çözümlerimiz ise aşağıda listelenmiştir:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Grafik-1’den hareketle bu noktaların grafik yöntemiyle çözdüğümüz noktalar ile birebir aynı olmamasının sebebi MiniZinc programının uygun çözüm alanı içerisini taramasından kaynaklanmaktadır. Program, bir uç noktadan başlayıp alan içerisini taramaya başlar. Daha sonra optimal çözümü bulduğu anda diğer noktalara bakmadan en son sonucu ekranda gösterir. Çünkü program sadece uç noktalarla ilgilenmemektedir. Bizim optimal çözümümüz o alan içinde olduğu için program da amacına uygun olarak hareket etmektedir. Oysaki grafik yöntemi, yalnız bulduğu uç noktaları ele almaktadır. Zaten bizim burada odaklanmamız gereken nokta da optimal çözümdür.

Çıktı ekranında görüldüğü gibi uygun çözümler “---------“ ile ayrılmıştır. “=========” ifadesi programın optimal çözüme ulaştığını belirtmektedir. Buna göre optimal çözümümüz

olarak bulunmuştur.

Soru-2’nin MiniZinc ile Çözümü:

2. örneğimizde ise iki tip ayakkabı mevcuttur. , 1. tip ayakkabıyı; ise 2. tip ayakkabıyı temsil etmektedir. Bu değişkenleri MiniZinc’te tanımlayacak olursak:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-2’deki genel kısıtlarımız:

Bu kısıtları ekledikten sonra MiniZinc’teki görünüm aşağıdaki gibi olacaktır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-2’deki amaç fonksiyonumuz olarak tanımlanmıştı. Bu amaç fonksiyonumuzu MiniZinc’e aktaralım ve çıktı komutumuzu ekleyelim:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Elde ettiğimiz uygun çözümlerimiz ise aşağıdaki gibi olacaktır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Görüldüğü gibi yalnız sonucu bulunmuştur. Bunun sebebi Soru-1’de açıkladığımız gibi uygun çözüm alanının tamamını taramamasından kaynaklanmaktadır.

Soru-3’ün MiniZinc ile Çözümü:

3. örneğimizde Giapetto oyuncak asker ve oyuncak tren yapmak istemektedir. , bir haftada üretilen asker sayısını; ise bir haftada üretilen tren sayısını temsil etmektedir. Bu değişkenleri programa ekleyecek olursak:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-3’deki genel kısıtlarımız:

Bu kısıtlarımızın MiniZinc’teki görünümü aşağıdaki gibi olacaktır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-3’deki amaç fonksiyonumuz olarak tanımlanmıştı. Bu amaç fonksiyonumuzu MiniZinc’e aktaralım ve çıktı komutunu ekleyelim:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Uygun çözümlerimiz ise:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Görüldüğü üzere optimal çözümümüzü grafik yöntemiyle yaptığımız şekilde olarak bulmuştur.

Soru-4’ün MiniZinc ile Çözümü:

Son sorumuzda ise bir elbise dikim firması, fabrika işçileri için iki tip üniforma üretmek istemektedir. , 1. tip elbiseyi; ise 2. tip elbiseyi belirtmektedir. Bu değişkenleri MiniZinc programına ekleyelim:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-4’deki genel kısıtlarımız:

Bu kısıtlarımızı da MiniZinc programına ekleyelim:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Soru-4’deki amaç fonksiyonumuz ’dir. Bu amaç fonksiyonumuzu da MiniZinc’e aktaralım ve çıktı komutumuzu ekleyelim:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Uygun çözümlerimiz ise aşağıdaki gibi olacaktır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Son sorumuz olan Soru-4’ün optimal sonucu ise olarak bulunmuştur.

Soru-5’in MiniZinc ile Çözümü:

Soru-5’te yer alan değişkenlerimizi, kısıtlarımızı ve amaç fonksiyonumuzu ve MiniZinc’e aktaralım ve çıktıyı gösterecek komutumuzu(output) ekleyelim:

**ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Kodumuzu çalıştıralım ve sonucu görelim.

**ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

Programda da görüldüğü üzere Soru-5’de ortak çözüm alanı olmadığı için ekrana “UNSATISFIABLE” sonucunu vermektedir.

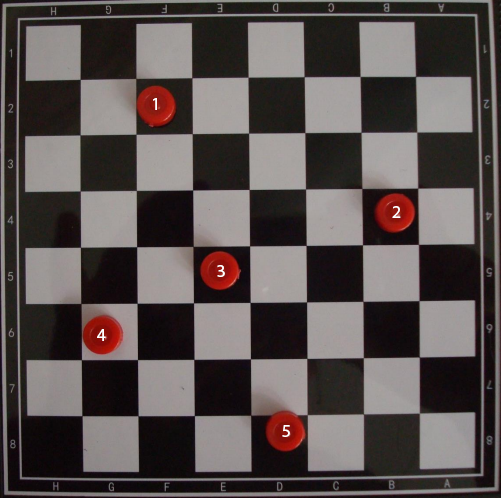
Soru 6:

Bu soruda ise yöneylem araştırması ve teorik bilgisayar bilimi alanlarında incelenen bir "kombinatorik optimizasyon" problemi olan gezgin satıcı problemini ele alacağız. Gezgin satıcı probleminin ne olduğuna değinmek gerekirse problem şu şekilde tanımlanabilir:

* Bir seyyar satıcı var.
* Bu satıcı, mallarını n şehirde satmak istiyor.
* Öte yandan, mantıklı bir şekilde, bu satıcı bu şehirleri mümkün olan en kısa şekilde ve her bir şehre maksimum bir kere uğrayarak turlamak istiyor.

Problemin amacı, satıcıya bu en kısa yolu sunabilmektir.

Bu problem 5 şehre indirgendiğinde ve satranç tahtası üzerinde ele alındığında:



Satranç tahtası üzerindeki kırmızı taşların her biri şehirleri, siyah-beyaz kareler ise yolları temsil etmektedir. 5 şehrimizin olduğunu ve taşların uzaklıklarını matris olarak MiniZinc’te tanımlayalım:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada şehir sayısını, alttaki matris değerleri ise taşın taşa olan uzaklığını temsil etmektedir(: 1. taşın 2. taşa olan uzaklığı).

Öncelikle şehir sayısını temsil etmek için “” değişkenini ve şehirler arası uzaklık matrisini tanımlayalım:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bir sonraki şehri ziyaret edebilmek için şu kod satırını yazmak gerekir:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Alınan toplam mesafe için aşağıdaki uzaklık değişkenini yani amaç fonksiyonunu tanımlayalım:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tüm şehirlere maksimum 1 kere gidilebilmesi için; yani aynı şehre tekrar uğramaması için aşağıdaki komut satırı yazılmalıdır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Görüldüğü gibi kısıtımız katedilen yoldan oluşmaktadır. Dolayısıyla ifademiz “constraint” komutu ile yazılmıştır. “circuit” fonksiyonu ile yazılmasının sebebi, bu yol bir daire gibi düşünüldüğünde gidilen bir şekilde tekrar gidilmeyeceğinden, bu daire en küçük olacak şekilde optimize edilmelidir. Dolayısıyla bu komut satırı ile -tıpkı bir daire gibi- her şehre maksimum 1 defa gidilmiş ve aynı şehre tekrar gidilmemiş olur.

Son olarak amaç fonksiyonunu minimum yapacak komutu yazalım ve çıktıyı ekrana gösterecek komutu da ekleyerek sonuçları görelim:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Sonuçlar aşağıdaki gibidir:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Görüldüğü üzere sırasıyla [4, 1, 5, 3, 2] şehirlerine gidildiğinde en kısa yoldan(24) şehirler ziyaret edilmiş olur.

Soru 7:

Son soruda ise klasik optimizasyon problemlerinden bir diğeri olan çanta problemi(knapsack problem) ele alınacaktır. Çanta problemi, basitçe bir çantanın içerisine maksimum fiyatta en fazla eşya konulması hedeflenir. Örnek olarak elimizde aşağıdaki eşyalar bulunsun:

1. Saat 200g 100TL

2. Fotoğraf makinesi 500g 300TL

3. Kamera 700g 700TL

4. Cep telefonu 300g 500TL

5. Anahtar 100g 10TL

Problemimizin amacı, 1000g kapasiteli çantaya yukarıdaki eşyaları maksimum fiyatta en fazla eşya konulacak şekilde yerleştirilmesidir.

Çözümümüz ise ürünler arasından kamera ve cep telefonunu seçtiğimizde 1200 TL ile maksimum değeri elde etmiş oluruz. Şimdi bu problemi MiniZinc programında tanımlayalım:

İlk adımda her zamanki gibi değişkenlerimizi tanımlayalım:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bizim burada tek kısıtımız olan ağırlık kısıtı, çanta kapasitesini geçmeyeceği için bu komut satırı aşağıdaki gibi yazılmalıdır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu adımda ise ağırlık kısıtı haricinde maksimum fiyatı elde etmek istediğimiz için bu komut satırı ve çıktı komutu aşağıdaki gibi olmalıdır:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Program çalıştırıldığında bizden parametre değerleri istenecektir:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Sonuçlarımız aşağıdaki gibidir:

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Görüldüğü üzere daha önce çözdüğümüz gibi “kamera” ve “telefon” sonuçları elde edilmiştir.

# YARARLANILAN KAYNAKLAR

URL, “MiniZinc”, <https://www.minizinc.org/>, (2014).

URL, “Optimizasyona Giriş ve Temel Kavramlar”,<http://ikucukkoc.baun.edu.tr/lectures/EMM3208/EMM3208W2.pdf>, (2018).

URL, “Optimizasyon Nedir”, <https://www.ceyrekmuhendis.com/optimizasyon-nedir/>, 02.03.2019).

URL, “Optimizasyon”, <https://tr.wikipedia.org/wiki/Optimizasyon>, (11.04.2020).

URL, “Doğrusal Programlama”, <http://www.ktu.edu.tr/dosyalar/ormanamenajmani_4ef1e.pdf>,(2018).

URL, “Doğrusal Programlama”, <http://www.ozyazilim.com/ozgur/marmara/karar/simplex.htm>, (1997).

URL, “Grafik Yöntemi ve Örnekler”, <http://www.ktu.edu.tr/dosyalar/ormanamenajmani_1a08b.pdf>,(2018).

URL, “Gezgin Satıcı Problemi”, https://tr.wikipedia.org/wiki/Gezgin\_sat%C4%B1c%C4%B1\_problemi, (19.04.2020)

URL,”Knapsack Problemi”, <http://bilgisayarkavramlari.sadievrenseker.com/2008/03/24/torba-problemi-knapsack-problem/,(24.03.2008)>.