ALTIN TOPLAMA OYUNU

KABA KOD

İçindekiler

Tüm Oyuncular Hareket Etme Kaba Kod

Not: Oyuncu önce yatay düzlemde ardından dikey düzlemde hareket etmektedir.

1. Başla.
2. Eğer altın haritasında bulunan altın sayısı büyükse 0 dan.
    1. Eğer oyuncu hedef belirlememiş ise
        1. Oyuncunun hedef belirlemesi için **SearchForGold** metodunu çağır.
    2. Eğer oyuncu hedef belirlemiş ancak belirlenen hedef alınmış ise
        1. Oyuncunun hedef bilgilerini sıfırla ve ardından **SearchForGold** metodunu çağır.
    3. Hedeflenen koordinatın Y ve X koordinatlarını değişkenlere ata. **targetY**,**targetX**
    4. Hedef ile oyuncu konumu arasındaki, Y ve X koordinatındaki uzaklık (kaç kare) olduğunu belirle ve değişkenlere bu değerleri ata. **targetYToPlayerY**,**targetXToPlayerX** (hesaplanan sonuç negatif bir değer ise koordinat küçülür, pozitif bir değer ise koordinat büyür)
    5. Oyuncunun tek bir hamlede kaç adım atacağını yeni bir değişkene ekle. **totalMoveLenght**
    6. Oyuncunun bulunduğu son konumu değişkenlere aktar. **tempCordY**,**tempCordX**
    7. Yatayda gitmesi gereken hareket, bir turda yapabileceği maksimum hareket sayısından küçük veya eşit ise
        1. Oyuncunun **tempCordX** koordinatını hedefin X koordinatı (**targetX**) olarak belirle.
        2. Oyuncunun toplamda atacağı hareket miktarından bu hamlede yaptığı hareket miktarını çıkart.
    8. Yatayda gitmesi gereken hareket, bir turda yapabileceği maksimum hareket sayısından büyük ise
        1. Hedef altına hareket pozitif yönde ise
            1. Oyuncunun **tempCordX** değerine oynayabileceği tüm adım sayısını (**totalMoveLenght**) ekle
        2. Hedef altına hareket pozitif yönde ise
            1. Oyuncunun **tempCordX** değerine oynayabileceği tüm adım sayısını (**totalMoveLenght**) çıkart
    9. Üzerinden geçtiği gizli altınları görünür yapması için **PrivateGoldShow()** metodunu çağır.
    10. Oyuncunun X koordinatını **UpdateCord()** metodu ile güncelle
    11. Yatayda yapılan hareketten sonra kalan adım var ise veya yatayda hiç hareket edilmemiş ise dikeyde hareket yap
    12. **totalMoveLenght** değeri 0dan büyük ise
    13. Dikeyde gitmesi gereken hareket, bir turda yapabileceği maksimum hareket sayısından küçük veya eşit ise
        1. Oyuncunun **tempCordY** koordinatını hedefin Y koordinatı (**targetY**) olarak belirle.
        2. Oyuncunun toplamda atacağı hareket miktarından bu hamlede yaptığı hareket miktarını çıkart.
    14. Yatayda gitmesi gereken hareket, bir turda yapabileceği maksimum hareket sayısından büyük ise
        1. Hedef altına hareket pozitif yönde ise
            1. Oyuncunun **tempCordY** değerine oynayabileceği tüm adım sayısını (**totalMoveLenght**) ekle
        2. Hedef altına hareket pozitif yönde ise

        1. Oyuncunun **tempCordY** değerine oynayabileceği tüm adım sayısını (**totalMoveLenght**) çıkart
    15. Üzerinden geçtiği gizli altınları görünür yapması için **PrivateGoldShow()** metodunu çağır.
    16. Oyuncunun Y koordinatını **UpdateCord()** metodu ile güncelle
    17. Hedefe ulaşıldı mı? Oyuncunun Y ve X koordinatları hedef ile eşit ise
        1. Oyuncunun kazandığı altını puanını oyuncuya ekle
        2. Kazanılan altın verisine hedefdeki altının değerini ekle
        3. Altını, altın haritasından sil
        4. Oyuncunun hedefleme bilgileri -1 olarak güncellenir
        5. Oyuncunun hedefe ulaştığına ait log kaydı oluştur.
    18. Hedefe ulaşılamadıysa bilgileri güncelle
        1. Oyuncunun kalan adım sayısını güncelle
        2. Oyuncunun kasasından hareket maliyetini çıkartma
        3. Oyuncunun hedefe ulaşması için kaç hamleye ihtiyacı olduğuna ait log kaydı oluştur.
3. Oyuncunun hamle sayısını 1 arttır.
4. Bitir.

Tüm Oyuncular Üzerinden Geçtikleri Altını Açma Kaba Kod

Yapınlan hareketin sonucunda gidilen yol üzerinde gizli altın bulunuyor ise onu görünür hale geririr.

cord: Hangi düzlemde hareket yapılacak

move: Kaç kare hareket edilecek

1. Başla.
2. Metoda gelen cord parametresi “X” değerine eşit ise
    1. hareket yönü negatif ise
        1. Oyuncunun konumundan itibaren **move** değeri kadar gizli altın haritasında geriye doğru git.
        2. Eğer gizli altın varsa
            1. Gizli altın haritasından kaldır.
            2. Normal altın haritasına ekle.
    2. hareket yönü pozitif ise
        1. Oyuncunun konumundan itibaren **move** değeri kadar gizli altın haritasında ileriye doğru git.
        2. Eğer gizli altın varsa
            1. Gizli altın haritasından kaldır.
            2. Normal altın haritasına ekle.

3. Metoda gelen cord parametresi “Y” değerine eşit ise
    1. hareket yönü negatif ise
        1. Oyuncunun konumundan itibaren **move** değeri kadar gizli altın haritasında geriye doğru git.
        2. Eğer gizli altın varsa
            1. Gizli altın haritasından kaldır.
            2. Normal altın haritasına ekle.
    2. hareket yönü pozitif ise
        1. Oyuncunun konumundan itibaren **move** değeri kadar gizli altın haritasında ileriye doğru git.
        2. Eğer gizli altın varsa
            1. Gizli altın haritasından kaldır.
            2. Normal altın haritasına ekle.
4. Bitir.

A Oyuncusu Hedef Belirleme Kaba Kod

1. Başla
2. Hedeflenen en yakın altın koordinatını en büyük integer değere eşitle.
3. Hedeflenen en yakın altının uzaklığını en büyük integer değere eşitle.
4. Hedeflenen en yakın altının değerini en küçük integer değere eşitle.
5. Parametre olarak gelen Map sınıfından altın matrisini al. (**GetGoldMap()**)
6. Altın matrisinin y koordinatının da dön (for)
    1. Altın matrisinin x koordinatının da dön (for)
        1. Altın matrisinde ki, for döngüsünden oluşan y ve x koordinatın daki altın miktarı 0a eşit değilse
        2. Altının konumunun A’ya olan uzaklığını hesapla.
            1. A’ya en yakın olan altın değerini hafızada tut.
7. Hedefe kaç adımda ulaşabileceğini hesapla
8. Hedefe ulaşacağı adımı tutan değişkene **SetRemainingSteps()** ile atama yap
9. Hedefin koordinat bilgilerini tutan değişkene **SetTargetedGoldCord()** ile atama yap
10. Hedefin altın miktarını tutan değişkene **SetTargetedGoldValue()** ile atama yap
11. Hedefe ulaştığında elde edeceği altın miktarını tutan değişkene **SetGoldEarnedOnReachTarget()** ile atama yap
12. Oyuncunun kasasından hedef belirleme maliyetini düş. **UpdatePlayerGoldValue()**

13. Oyuncunun oyun boyunca harcadığı toplam altın miktarına arama maliyetini **SetTotalAmountOfGoldSpent(this.GetSearchCost())** ekle.
14. Oyuncunun hareket geçmişine gerekli metinleri **SetLog()** ile ekle.
15. Oyuncunun hedeflediği konumu Map sınıfından metoda gelen map parametresine **map.SetPlayerTarget()** ile aktar.
16. Oyuncunun hedeflediği konuma kalan adım sayısını Map sınıfından metoda gelen map parametresi **map.SetPlayerRemainingSteps()** ile aktar
17. Bitir.

B Oyuncusu Hedef Belirleme Kaba Kod

1. Başla
2. Hedeflenen en yakın altın koordinatını en büyük integer değere eşitle. (**nearestGoldY,nearestGoldX**). **Başlangıç değerleri 2147483647**
3. Hedeflenen altından elde edilen kar. (**nearestGoldProfit**). **Başlangıç değerleri -2147483647**
4. Hedeflenen altına ulaşmak için gereken tur sayısı (**remainingSteps**). **Başlangıç değerleri -2147483647**
5. Hedeflenen altının değeri (**nearestGoldValue**).**Başlangıç değerleri -2147483647**
6. Hedeflenen altına giden yolun uzunluğu (**nearestGoldPathLength**) **Başlangıç değerleri 2147483647**
7. Parametre olarak gelen Map sınıfından altın matrisini al. (**GetGoldMap()**) oluşturulan matrisin adı **goldArray**
8. Altın matrisinin y koordinatının da dön (for)
    1. Altın matrisinin x koordinatının da dön (for)
        1. Altın matrisinde ki, for döngüsünden oluşan y ve x koordinatındaki altın miktarı 0a eşit değilse
            1. Geçici olarak hedefin kaç kare uzaklıkta olduğunu tutan bir değişken oluştur (**tempPathLength**) ve uzaklık değerini bu değişkene aktar
            2. Geçici olarak hedefin kaç adım uzaklıkta olduğunu tutan bir değişken oluştur (**tempRemainingSteps**) ve uzaklık değerini hesapla ve değişkene aktar.
            3. Geçici olarak hedefe ulaştığında kazanacağı toplam altın miktarını hesapla ve (**tempProfit**) değişkene aktar.
            4. Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) büyükse veya eşitse
                1. **tempPathLength** değeri hedeflenen altın uzak değerinden küçükse (**nearestGoldPathLength**) **VE** Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) eşitse
                    1. Aşağıda belirtilen değişkenleri güncelle.
                    2. nearestGoldPathLength = tempPathLength;
                    3. remainingSteps = tempRemainingSteps;
                    4. nearestGoldProfit = tempProfit;

5. nearestGoldY = goldY;
6. nearestGoldX = goldX;
7. nearestGoldValue = goldArray[goldY, goldX];

2. Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) büyükse
    1. Aşağıda belirtilen değişkenleri güncelle.
    2. nearestGoldPathLength = tempPathLength;
    3. remainingSteps = tempRemainingSteps;
    4. nearestGoldProfit = tempProfit;
    5. nearestGoldY = goldY;
    6. nearestGoldX = goldX;
    7. nearestGoldValue = goldArray[goldY, goldX];

9. Hedefe ulaşacağı adımı tutan değişkene **SetRemainingSteps()** ile atama yap
10. Hedefin koordinat bilgilerini tutan değişkene **SetTargetedGoldCord()** ile atama yap
11. Hedefin altın miktarını tutan değişkene **SetTargetedGoldValue()** ile atama yap
12. Hedefe ulaştığında elde edeceği altın miktarını tutan değişkene **SetGoldEarnedOnReachTarget()** ile atama yap
13. Oyuncunun kasasından hedef belirleme maliyetini düş. **UpdatePlayerGoldValue()**
14. Oyuncunun oyun boyunca harcadığı toplam altın miktarına arama maliyetini **SetTotalAmountOfGoldSpent(this.GetSearchCost())** ekle.
15. Oyuncunun hareket geçmişine gerekli metinleri **SetLog()** ile ekle.
16. Oyuncunun hedeflediği konumu Map sınıfından metoda gelen map parametresine **map.SetPlayerTarget()** ile aktar.
17. Oyuncunun hedeflediği konuma kalan adım sayısını Map sınıfından metoda gelen map parametresi **map.SetPlayerRemainingSteps()** ile aktar
18. Bitir.

C Oyuncusu Hedef Belirleme Kaba Kod

1. Başla.
2. C oyuncusunun kendine yakın gizli altınları açma özelliğini aktif etmek için **PrivateGoldShow(map)** metodunu çağır.
3. Hedeflenen en yakın altın koordinatını en büyük integer değere eşitle. (**nearestGoldY,nearestGoldX**). **Başlangıç değerleri 2147483647**
4. Hedeflenen altından elde edilen kar. (**nearestGoldProfit**). **Başlangıç değerleri -2147483647**
5. Hedeflenen altına ulaşmak için gereken tur sayısı (**remainingSteps**). **Başlangıç değerleri -2147483647**

6. Hedeflenen altının değeri (**nearestGoldValue**).**Başlangıç değerleri -2147483647**
7. Hedeflenen altına giden yolun uzunluğu (**nearestGoldPathLength**) **Başlangıç değerleri 2147483647**
8. Parametre olarak gelen Map sınıfından altın matrisini al. (**GetGoldMap()**) oluşturulan matrisin adı **goldArray**
9. Altın matrisinin y koordinatının da dön (for)
    1. Altın matrisinin x koordinatının da dön (for)
        1. Altın matrisinde ki, for döngüsünden oluşan y ve x koordinatlı altın miktarı 0a eşit değilse
            1. Geçici olarak hedefin kaç kare uzaklıkta olduğunu tutan bir değişken oluştur (**tempPathLength**) ve uzaklık değerini bu değişkene aktar
            2. Geçici olarak hedefin kaç adım uzaklıkta olduğunu tutan bir değişken oluştur (**tempRemainingSteps**) ve uzaklık değerini hesapla ve değişkene aktar.
            3. Geçici olarak hedefe ulaştığında kazanacağı toplam altın miktarını hesapla ve (**tempProfit**) değişkene aktar.
            4. Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) büyükse veya eşitse
                1. **tempPathLength** değeri hedeflenen altın uzak değerinden küçükse (**nearestGoldPathLength**) **VE** Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) eşitse
                    1. Aşağıda belirtilen değişkenleri güncelle.
                    2. nearestGoldPathLength = tempPathLength;
                    3. remainingSteps = tempRemainingSteps;
                    4. nearestGoldProfit = tempProfit;
                    5. nearestGoldY = goldY;
                    6. nearestGoldX = goldX;
                    7. nearestGoldValue = goldArray[goldY, goldX];
                2. Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) büyükse
                    1. Aşağıda belirtilen değişkenleri güncelle.
                    2. nearestGoldPathLength = tempPathLength;
                    3. remainingSteps = tempRemainingSteps;
                    4. nearestGoldProfit = tempProfit;
                    5. nearestGoldY = goldY;
                    6. nearestGoldX = goldX;
                    7. nearestGoldValue = goldArray[goldY, goldX];
10. Hedefe ulaşacağı adımı tutan değişkene **SetRemainingSteps()** ile atama yap
11. Hedefin koordinat bilgilerini tutan değişkene **SetTargetedGoldCord()** ile atama yap
12. Hedefin altın miktarını tutan değişkene **SetTargetedGoldValue()** ile atama yap

13. Hedefe ulaştığında elde edeceği altın miktarını tutan değişkene **SetGoldEarnedOnReachTarget()** ile atama yap
14. Oyuncunun kasasından hedef belirleme maliyetini düş. **UpdatePlayerGoldValue()**
15. Oyuncunun oyun boyunca harcadığı toplam altın miktarına arama maliyetini **SetTotalAmountOfGoldSpent(this.GetSearchCost())** ekle.
16. Oyuncunun hareket geçmişine gerekli metinleri **SetLog()** ile ekle.
17. Oyuncunun hedeflediği konumu Map sınıfından metoda gelen map parametresine **map.SetPlayerTarget()** ile aktar.
18. Oyuncunun hedeflediği konuma kalan adım sayısını Map sınıfından metoda gelen map parametresi **map.SetPlayerRemainingSteps()** ile aktar
19. Bitir.

C Oyuncusu Gizli Altınları Açma Kaba Kod

1. Başla.
2. Metot başladığından itibaren kaç adet gizli altın açtığını belirlemek için **control** adında bir değişken oluştur. Değişkenin başlangıç değerini 0 olarak belirle.
3. control değişkeni C oyuncunun bir turda açabileceği maksimum gizli altın değerinden büyük oluncaya dek.
    1. Hedeflenen en yakın gizli altın koordinatını en büyük integer değere eşitle. (**nearestPrivateGoldY,nearestPrivateGoldX**). **Başlangıç değerleri 2147483647**
    2. Hedeflenen gizli altına giden yolun uzunluğu (**nearestPrivateGoldPathLength**) **Başlangıç değerleri 2147483647**
    3. Hedeflenen gizli altının değeri (**nearestPrivateGoldValue**) değer olarak **-1** değerini ata
    4. Gizli altın haritasında bulunan gizli altın miktarı 0dan büyükse
        1. Parametre olarak gelen Map sınıfından gizli altın matrisini al. (**GetPrivateGoldMap()**) oluşturulan matrisin adı **goldArray**
        2. Gizli altın matrisinin y koordinatında dön (for)
            1. Gizli altın matrisinin x koordinatında dön (for)
                1. Gizli altın matrisinde ki, for döngüsünden oluşan y ve x koordinatlı gizli altın miktarı 0a eşit değilse
                2. Geçici olarak hedefin kaç kare uzaklıkta olduğunu tutan bir değişken oluştur (**tempPathLength**) ve uzaklık değerini bu değişkene aktar
                3. **tempPathLength** değeri hedeflenen gizli altının uzaklık değerinden küçükse (**nearestPrivateGoldPathLength**)
                    1. Aşağıda belirtilen değişkenleri güncelle.
                    2. nearestPrivateGoldPathLength = tempPathLength;
                    3. nearestPrivateGoldY = goldY;
                    4. nearestPrivateGoldX = goldX;

5. nearestPrivateGoldValue = goldArray[goldY, goldX];

5. Gizli altın haritasında bulunan gizli altın miktarı 0dan eşitse
    1. Döngüden çık.
6. Parametre olarak gelen Map nesnesinin metotlarından **UpdateGoldMapPoint()** çağır ve belirlenen koordinatlardaki gizli altını, altın haritasına eklenecektir
7. Parametre olarak gelen Map nesnesinin metotlarından **RemovePrivateGoldPoint()** çağır ve belirlenen koordinatlardaki gizli altını, gizli altın matrisinden sil.
8. Oyuncunun hareket geçmişine gerekli metinleri **SetLog()** ile ekle.
9. Control değişkeninin değerini bir arttır.
10. Bitir.

D Oyuncusu Hedef Belirleme Kaba Kod

1. Başla
2. Hedeflenen en yakın altın koordinatını en büyük integer değere eşitle. (**nearestGoldY,nearestGoldX**). **Başlangıç değerleri 2147483647**
3. Hedeflenen altından elde edilen kar. (**nearestGoldProfit**). **Başlangıç değerleri -2147483647**
4. Hedeflenen altına ulaşmak için gereken tur sayısı (**remainingSteps**). **Başlangıç değerleri -2147483647**
5. Hedeflenen altının değeri (**nearestGoldValue**).**Başlangıç değerleri -2147483647**
6. Hedeflenen altına giden yolun uzunluğu (**nearestGoldPathLength**) **Başlangıç değerleri 2147483647**
7. Oyuncu hangi oyuncuya ait hedefi seçti. (**whoseTarget**) **Başlangıç değeri String.Empty**
8. Oyuncunun kendi koordinat bilgilerini (**thisCord**) bir diziye aktar. Bunun için player nesnesinin **GetLastCord()** metodunu kullanılacaktır.
9. Diğer oyuncuların hedeflerini map nesnesinin (**GetPlayerTarget**) metodunu kullanarak değişkenlerine aktar. **aPlayerTarget, bPlayerTarget, cPlayerTarget**
10. D'nin diğer oyuncuların hedeflerine ulaşması için gereken tur sayısını belirlemek için DPlayer sınıfına ait olan **GetStepsOfOtherPlayersTarget** metodunu çağır. Gelen değerleri gerekli değişkenlere aktar. **remainingSteps_ATarget ,remainingSteps_BTarget ,remainingSteps_CTarget**
11. A'nın hedefini hedef belirlemek için.
12. D'nin A'nın hedefine ulaşacağı hamle sayısı küçükse A'nın hedefine ulaşacağı hamle sayısından **VE** A'nın hedefine kalan adım sayısı -1 değerine eşit değilse
    1. Aşağıda belirtilen değişkenleri güncelle.
    2. whoseTarget = "A";
    3. nearestGoldY = aPlayerTarget[0];
    4. nearestGoldX = aPlayerTarget[1];

5. nearestGoldValue = map.GetGoldPointValue(aPlayerTarget[0], aPlayerTarget[1]);
6. remainingSteps = remainingSteps_ATarget;

13. B'nin hedefini hedef belirlemek için.
14. D'nin B'nin hedefine ulaşacağı hamle sayısı küçükse B'nin hedefine ulaşacağı hamle sayısından **VE** B'nin hedefine kalan adım sayısı -1 değerine eşit değilse
    1. Aşağıda belirtilen değişkenleri güncelle.
    2. whoseTarget = "B";
    3. nearestGoldY = bPlayerTarget[0];
    4. nearestGoldX = bPlayerTarget[1];
    5. nearestGoldValue = map.GetGoldPointValue(bPlayerTarget[0], bPlayerTarget[1]);
    6. remainingSteps = remainingSteps_ATarget;
15. C'nin hedefini hedef belirlemek için.
16. D'nin C'nin hedefine ulaşacağı hamle sayısı küçükse C'nin hedefine ulaşacağı hamle sayısından **VE** C'nin hedefine kalan adım sayısı -1 değerine eşit değilse
    1. Aşağıda belirtilen değişkenleri güncelle.
    2. whoseTarget = "C";
    3. nearestGoldY = cPlayerTarget[0];
    4. nearestGoldX = cPlayerTarget[1];
    5. nearestGoldValue = map.GetGoldPointValue(bPlayerTarget[0], bPlayerTarget[1]);
    6. remainingSteps = remainingSteps_ATarget;
17. D Oyuncusu diğer oyuncuların hedeflerine ulaşamıyorsa
    1. Parametre olarak gelen Map sınıfından altın matrisini al. (**GetGoldMap()**) oluşturulan matrisin adı **goldArray**
    2. Diğer oyuncuların hedeflerine ulaşamadığı için onları hedef olmaktan çıkartmamız gerekiyor bunun için altın matrisinin bir kopyasını oluşturalım. Oluşturulan matrise **tempGoldArray** ismini verelim
    3. A Oyuncusunun hedefi -1 değerine eşit değilse
        1. A Oyuncusunun hedeflediği koordinatlardaki altın değerini **tempGoldArray** matrisinde 0 olarak belirtelim.
    4. B Oyuncusunun hedefi -1 değerine eşit değilse
        1. B Oyuncusunun hedeflediği koordinatlardaki altın değerini **tempGoldArray** matrisinde 0 olarak belirtelim.
    5. C Oyuncusunun hedefi -1 değerine eşit değilse
        1. C Oyuncusunun hedeflediği koordinatlardaki altın değerini **tempGoldArray** matrisinde 0 olarak belirtelim.
    6. Diğer oyuncuların altınları kaldırıldığında oyunda altın kalmıyorsa eğer hedefleri haritaya geri ekleyelim.

7. Kopyasını oluşturduğumuz haritada kaç adet altın bulunduğunu belirlemek için **goldCount** adında bir değişken oluşturalım ve başlangıç değeri olarak 0 verelim.
8. Kopyasını oluşturduğumuz haritada kaç adet altın olduğunu hesaplayalım.
9. Kopyasını oluşturduğumuz haritadaki altın sayısı 0'a eşit ise
    1. Diğer oyuncuların hedeflerini haritaya tekrar dahil edelim.

18. Kopya Altın matrisinin y koordinatının da dön (for)
    1. Kopya Altın matrisinin x koordinatının da dön (for)
        1. Kopya Altın matrisinde ki, for döngüsünden oluşan y ve x koordinatındaki altın miktarı 0a eşit değilse
            1. Geçici olarak hedefin kaç kare uzaklıkta olduğunu tutan bir değişken oluştur (**tempPathLength**) ve uzaklık değerini bu değişkene aktar
            2. Geçici olarak hedefin kaç adım uzaklıkta olduğunu tutan bir değişken oluştur (**tempRemainingSteps**) ve uzaklık değerini hesapla ve değişkene aktar.
            3. Geçici olarak hedefe ulaştığında kazanacağı toplam altın miktarını hesapla ve (**tempProfit**) değişkene aktar.
            4. Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) büyükse veya eşitse
                1. **tempPathLength** değeri hedeflenen altın uzak değerinden küçükse (**nearestGoldPathLength**) **VE** Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) eşitse
                    1. Aşağıda belirtilen değişkenleri güncelle.
                    2. nearestGoldPathLength = tempPathLength;
                    3. remainingSteps = tempRemainingSteps;
                    4. nearestGoldProfit = tempProfit;
                    5. nearestGoldY = goldY;
                    6. nearestGoldX = goldX;
                    7. nearestGoldValue = tempGoldArray[goldY, goldX];
                2. Geçici olarak tutulan **tempProfit** değeri hedeflenen altın değerinden (**nearestGoldProfit**) büyükse
                    1. Aşağıda belirtilen değişkenleri güncelle.
                    2. nearestGoldPathLength = tempPathLength;
                    3. remainingSteps = tempRemainingSteps;
                    4. nearestGoldProfit = tempProfit;
                    5. nearestGoldY = goldY;
                    6. nearestGoldX = goldX;
                    7. nearestGoldValue = tempGoldArray[goldY, goldX];

19. Hedefe ulaşacağı adımı tutan değişkene **SetRemainingSteps()** ile atama yap

20. Hedefin koordinat bilgilerini tutan değişkene **SetTargetedGoldCord()** ile atama yap
21. Hedefin altın miktarını tutan değişkene **SetTargetedGoldValue()** ile atama yap
22. Hedefe ulaştığında elde edeceği altın miktarını tutan değişkene **SetGoldEarnedOnReachTarget()** ile atama yap
23. Oyuncunun kasasından hedef belirleme maliyetini düş. **UpdatePlayerGoldValue()**
24. Oyuncunun oyun boyunca harcadığı toplam altın miktarına arama maliyetini **SetTotalAmountOfGoldSpent(this.GetSearchCost())** ekle.
25. **whoseTarget** değer eşit değilse **String.Empty** değerini o zaman herhangi bir oyuncunun hedefine daha önce varacaktır.
    1. Oyuncunun hareket geçmişine gerekli metinleri **SetLog()** ile ekle.
26. **whoseTarget** değer eşit ise **String.Empty** değerine o zaman herhangi bir oyuncunun hedefine ondan daha önce varamayacaktır.
    1. Oyuncunun hareket geçmişine gerekli metinleri **SetLog()** ile ekle.
27. Oyuncunun hedeflediği konumu Map sınıfından metoda gelen map parametresine **map.SetPlayerTarget()** ile aktar.
28. Oyuncunun hedeflediği konuma kalan adım sayısını Map sınıfından metoda gelen map parametresi **map.SetPlayerRemainingSteps()** ile aktar
29. Bitir.