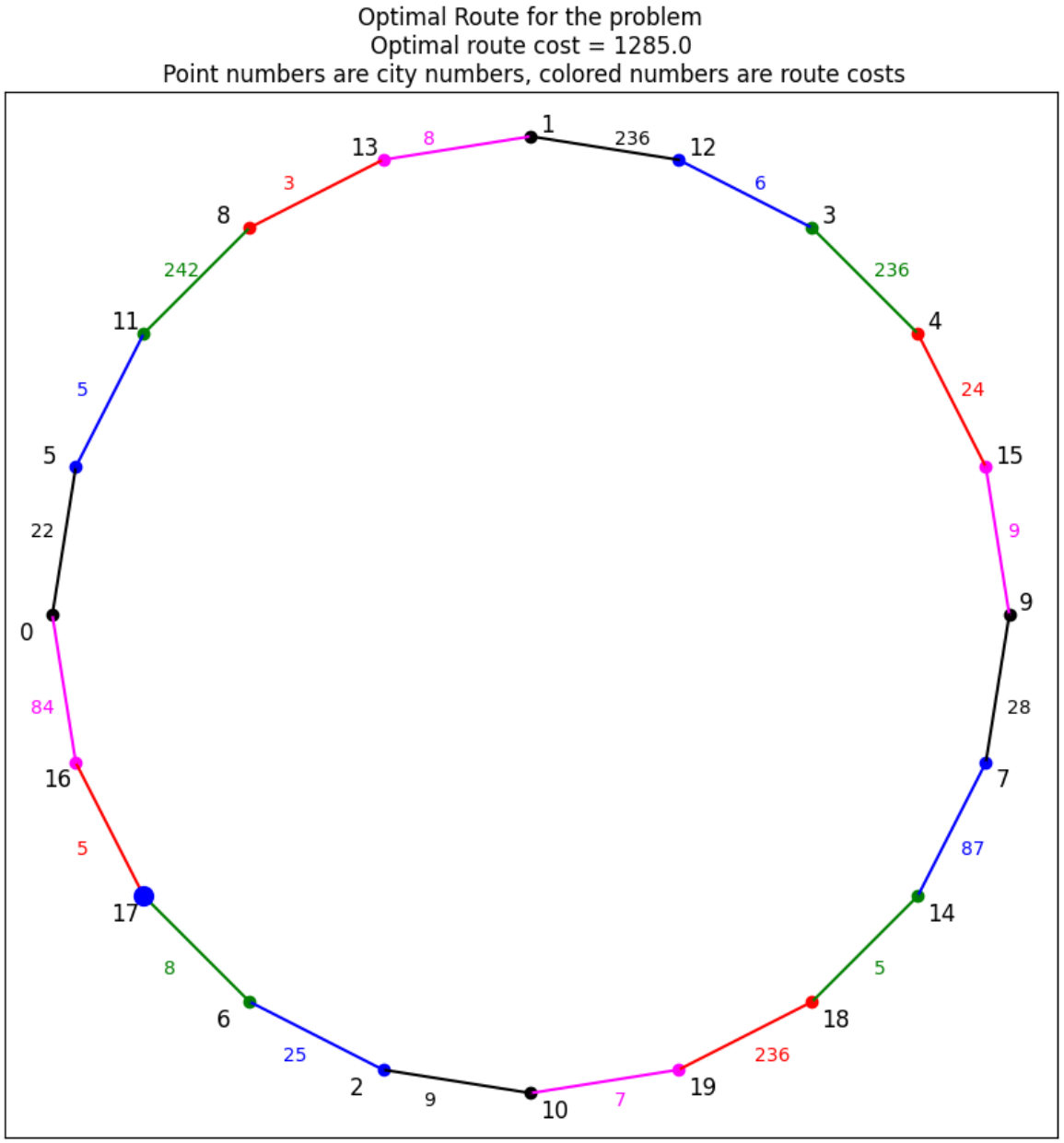
between 1 and 20. The cost of travelling to the next city for all routes is marked along the path.

In [ ]:
```
plotRoute2(routes[-1], x_coords, y_coords, costMatrix, costs[-1])
```



Exercise 8

13. Let $X_1, \ldots, X_n$ be independent and identically distributed random variables having unknown mean μ. For given constants $a < b$, we are interested in estimating $p = P\{a < \sum_{i=1}^{n} X_i/n - \mu < b\}$.

(a) Explain how we can use the bootstrap approach to estimate p.
(b) Estimate p if $n = 10$ and the values of the X_i are 56, 101, 78, 67, 93, 87, 64, 72, 80, and 69. Take $a = -5, b = 5$.

In the following three exercises $X_1, \ldots, X_n$ is a sample from a distribution whose variance is (the unknown) σ^2. We are planning to estimate σ^2 by the sample variance $S^2 = \sum_{i=1}^{n}(X_i - \overline{X})^2/(n-1)$, and we want to use the bootstrap technique to estimate $\mathrm{Var}(S^2)$.

a) CHANGED

Sample n numbers from the emperical distribution $X_1, \ldots X_n$ with replacement, do this a large amount of times. For each of these samples check if the relation holds, and from this find the probablity of a sample satisfying the relation. The mean μ can be preestimated, as just the mean of the samples we bootstrap from.

This can even be done a large amount of times, to get a confidence interval for p

b) CHANGED

When $n = 10$ sample 10×10000 from the empirical distribution, and find p. Here we do this 500 times to also get a 95% CI for p.

In [ ]:
```
Xs = np.array([56, 101, 78, 67, 93, 87, 64, 72, 80, 69])
mu = np.mean(Xs)

ps = []
m = 10000
k = 500

mu = np.mean(Xs)
a = -5
b = 5

for i in range(k):
    sample = rnd.choice(Xs, (len(Xs),m), replace = True)
    #mu = np.mean(np.mean(sample, axis= 0))
    sample = sample - mu
    me = np.mean(sample, axis = 0)

    p = np.mean((me > a) * (me < b))
    ps.append(p)
```

In [ ]:
```
alpha = 0.05
low = np.quantile(ps, alpha/2)
high = np.quantile(ps, 1 - alpha/2)
print(f"Mean probabilty = {round(np.mean(ps),3)*100}%")
print(f"95% CI: [{round(low,3)},{round(high,3)}]")
```

```
Mean probabilty = 76.6%
95% CI: [0.758,0.775]
```

15. If $n = 15$ and the data are

$$5, 4, 9, 6, 21, 17, 11, 20, 7, 10, 21, 15, 13, 16, 8$$

approximate (by a simulation) the bootstrap estimate of $\text{Var}(S^2)$.

UNCHANGED

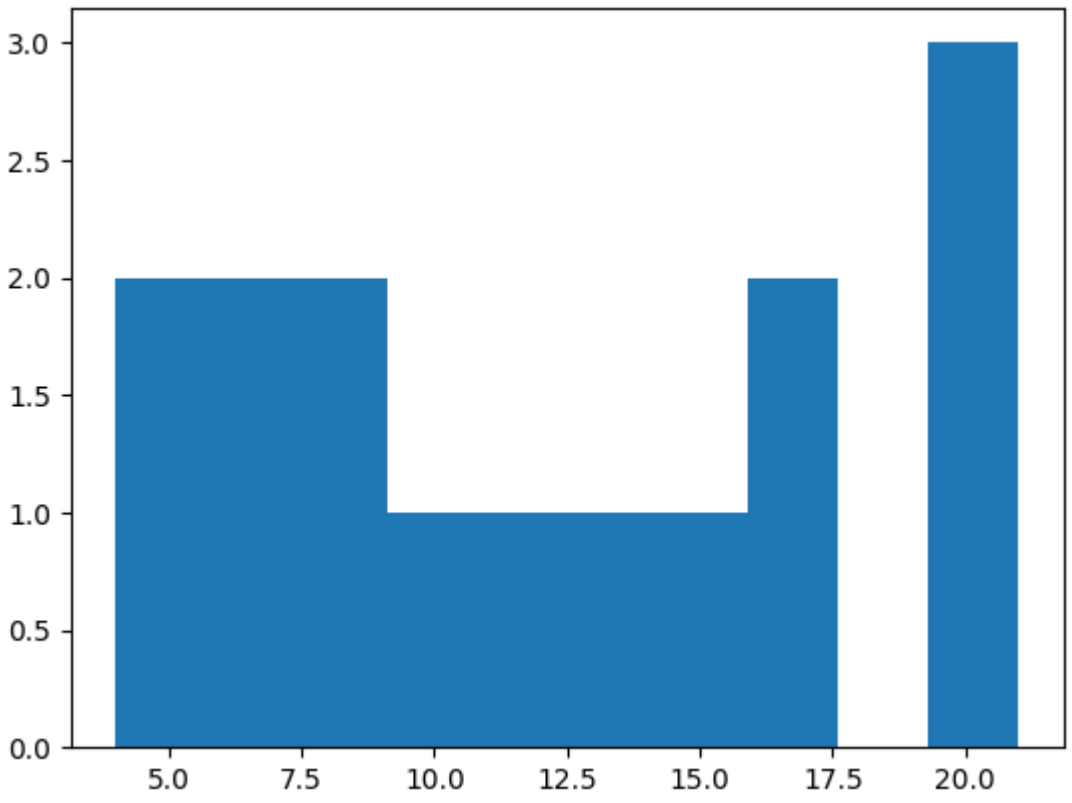
To find the variance of the sample variance we sample 15 ranom numbers from the dataset $m = 10000$ times. For each sample of 15 find it's variance. Then find the variance of the m estimated variances.

Notice the variance in a given sample is not very large, but the variance between samples varies a lot. This may be attributed to the emipircal distribution having large amounts of either relatively small numbers, and relatively large numbers. This means the variances between samples can change drastically.

In [ ]:
```
Xs = np.array([5, 4, 9, 6, 21, 17, 11, 20, 7, 10, 21, 15, 13, 16, 8])
m = 10000
vars = []
sample = rnd.choice(Xs, (len(Xs),m), replace = True)
var = np.var(sample, ddof= 1, axis = 0)

print("Variance in the sample variance is:", round(np.var(var, ddof=1),3))
_ = plt.hist(Xs, bins = 10)
```

```
Variance in the sample variance is: 60.145
```



3) CHANGED

Recall for the Pareto distribution that the mean of the sample is not well explained by the majority of points in the distribtution. This means it's difficult to sample enough points to accurately estimate the mean, resulting in a large variance in our estimate of it. This is not the case for the median value, which can be seen in it's low bootstrap variance. The median is more robust to what could be thought of as outliers. They are of course not really outliers, but in this case the few sampled large values, which so heavily sway the mean value, do not affect the median, since there's not a lot of them.

In [ ]:
```
def generatePareto(k, beta = 1, n = 10000):
    Us = rnd.rand(n)
    return beta * (Us**(-1/k))
beta = 1
k = 1.05
N = 200
r = 100
Xs = generatePareto(k, beta, N)
dist_mean = np.mean(Xs)
dist_median = np.median(Xs)

print(f"The mean of the generated variables is {round(dist_mean,3)}")
print(f"The median of the generated variables is {round(dist_median,3)}")
```

```
The mean of the generated variables is 5.89
The median of the generated variables is 1.644
```

In [ ]:
```
samples = rnd.choice(Xs, (len(Xs), r) , replace = True)
means = np.mean(samples, axis=0)
medians = np.median(samples, axis=0)

print(f"Bootstrap of the variance of sample mean: {round(np.var(means, ddof = 1)
print(f"Bootstrap of the variance of sample median: {round(np.var(medians, ddof
```

```
Bootstrap of the variance of sample mean: 2.08
Bootstrap of the variance of sample median: 0.017
```