Question 1: [10pts, HELP]: Write a function that takes an argument x and a hypothesis (however you represent it) and computes a size principle likelihood (e.g. where the likelihood of each number in the set is equal). Write down what likelihood each hypothesis assigns to each data point in it. What does each hypothesis assign to data points not in it?

In [40]:

```
#Calculating Hypotheses (ranges 1-100)
# h1 = even
# h2 = odd
# h3 = squares
# h4 = primes
# h5 = multiples of 5
# h6 = multiples of 10
# h7 = all nums

h1 = []
h2 = []
h3 = []
h4 = []
h5 = []
h6 = []
h7 = []

x = 1
i = 2

while x <= 100:
    if x%2 == 0:
        h1.append(x)
    else:
        h2.append(x)
    if x*x <= 100:
        h3.append(x*x)
    while i < x:
        if x%i == 0:
            break
        i += 1
    if i == x:
        h4.append(x)
    if x%5 == 0:
```

```
    if x%5 == 0:
        h5.append(x)
    if x%10 == 0:
        h6.append(x)
    h7.append(x)
    x += 1
    i = 2

print("h1 (evens):", h1)
print("h2 (odds):", h2)
print("h3 (squares):", h3)
print("h4 (primes):", h4)
print("h5 (5x):", h5)
print("h6 (10x):", h6)
print("h7 (all):", h7)
```

```
h1 (evens): [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22,
24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48,
50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74,
76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100]
h2 (odds): [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 2
3, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 4
9, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 7
5, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99]
h3 (squares): [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
h4 (primes): [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31
, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89
, 97]
h5 (5x): [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55,
60, 65, 70, 75, 80, 85, 90, 95, 100]
h6 (10x): [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
h7 (all): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26
, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39
, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52
, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65
, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78
, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91
, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```

In [41]:

```
def splike(data, hypo):
    # data = [1, 2, 3], hypo = primes
    # returns the size principle likelihood of numbers (x) in hypothesis
    likelihood = 1
    for x in data:
        if x in hypo:
            # if x is in hypothesis, increment likelihood by a multiple of 1/length(hypothesis)
            likelihood = likelihood * (1/len(hypo))
        else:
            # otherwise, likelihood becomes 0
            likelihood = 0
    return likelihood


data = [7]
print(splike(data, h4))
```

```
0.04
```

For each data point contained in a hypothesis, the likelihood assigned by the hypothesis is (1/size of hypothesis). -H1: 1/50 -H2: 1/50 -H3: 1/10 -H4: 1/25 -H5: 1/20 -H6: 1/10 -H7: 1/100

For each data point not contained in a hypothesis, the likelihood is 0.

Question 2

In [42]:

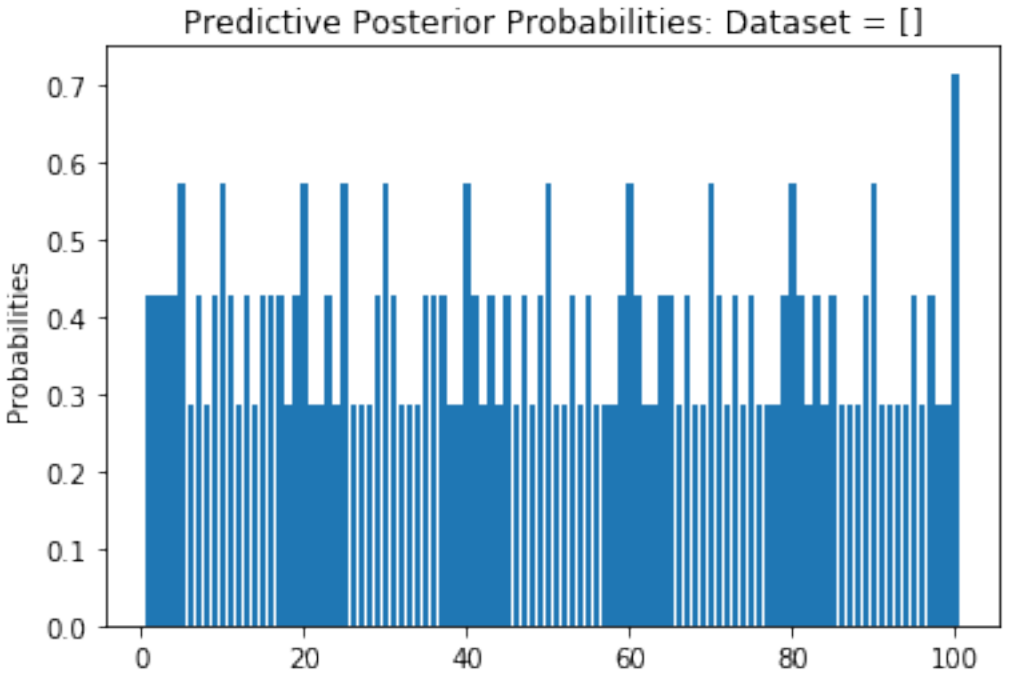
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define lists for our given datasets
data_a = []
data_b = [50]
data_c = [53]
data_d = [50, 53]
data_e = [16]
data_f = [10, 20]
data_g = [2, 4, 8]
data_h = [2, 4, 8, 10]
```

In [43]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_a

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)
```

```
for i in range(1, 101):

    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.14285714285714288, 0.14285714285714288, 0.1428571
4285714288, 0.14285714285714288, 0.14285714285714288
, 0.14285714285714288, 0.14285714285714288]
```



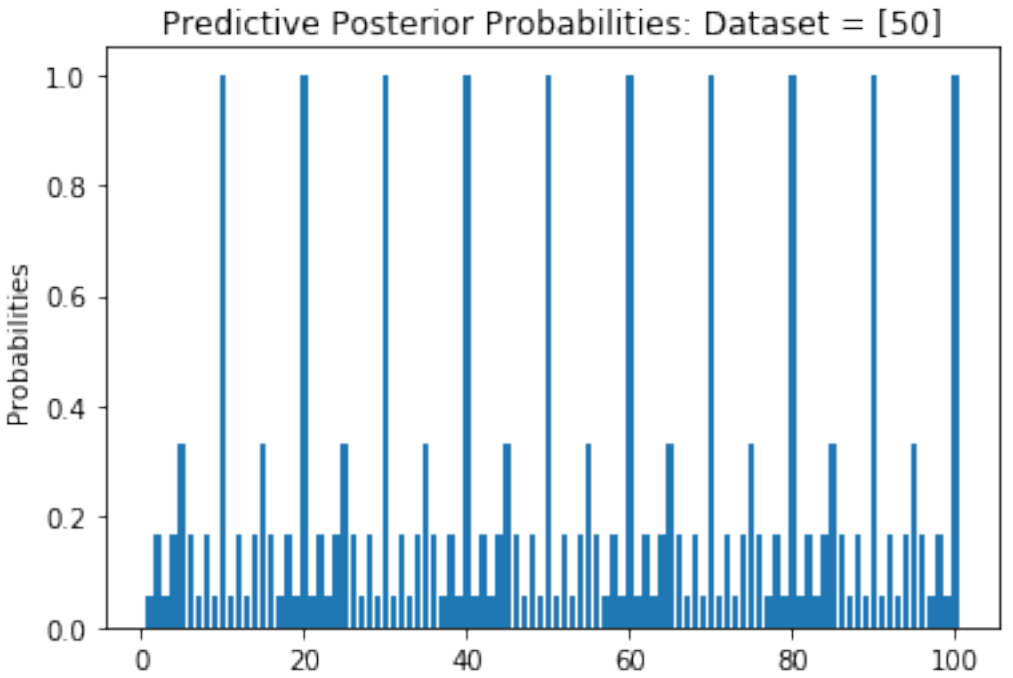
Explanation; Dataset = [] This graph shows higher probabilities for the integers between 1-100 that satisfy the most hypotheses. For example, the numbers which are multiples of 10 are also multiples of 5, and are even numbers, so they have higher probabilities on the graph since they satisfy more hypotheses than odd numbers or primes.

In [44]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_b

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
```

```
)

plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.11111111111111112, 0.0, 0.0, 0.0, 0.2777777777777
778, 0.5555555555555556, 0.05555555555555556]
```



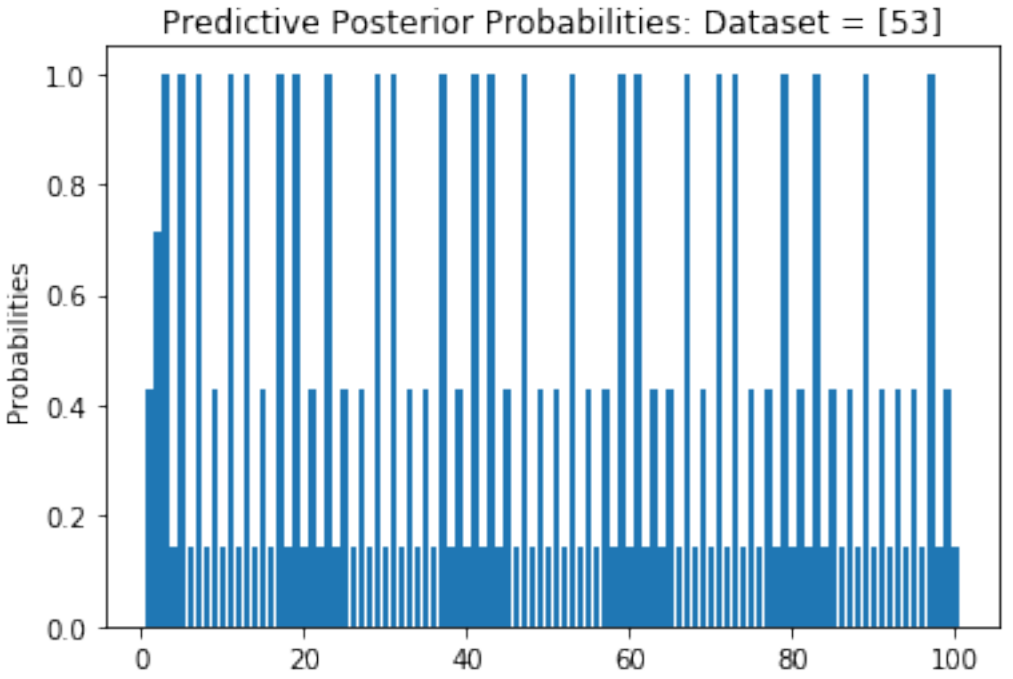
Explanation; Dataset = [50]: This graph matches my intuition as to what I expected, since every hypothesis that contains 50 also contains 10, 20, 30, ..., 90, 100. In addition, since 50 is also a multiple of 5, multiples of 5 are fairly well represented on this graph as well.

In [45]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_c
```

```
# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.0, 0.2857142857142857, 0.0, 0.5714285714285714, 0
.0, 0.0, 0.14285714285714285]
```



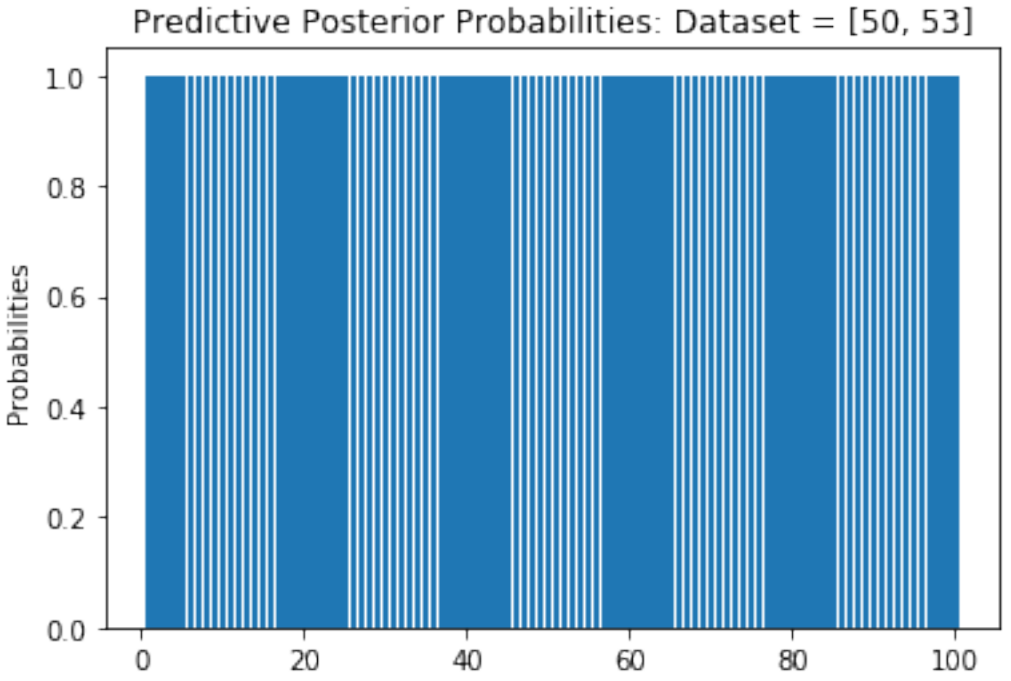
Explanation; Dataset = [53]: This graph matches my intuition because the number 53 is prime, and primes are well represented in this graph (aside from 2, which is a unique prime since it is also an even number).

In [46]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_d

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
```

```
    likelihood.append(like * prior)


for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0]
```



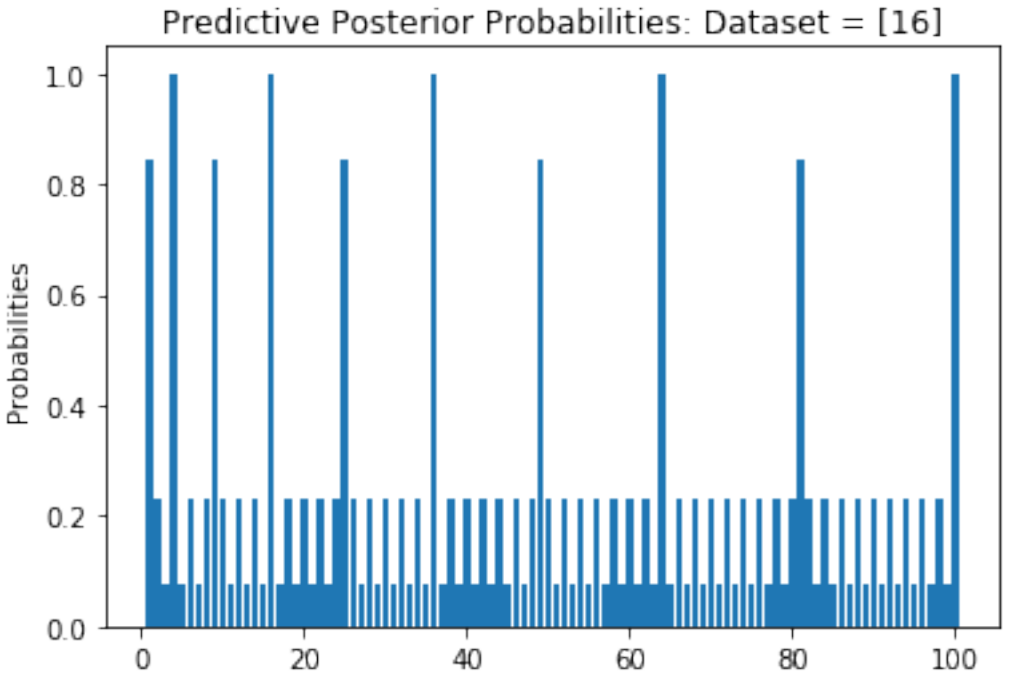
Explanation; Dataset = [50, 53]: Although I originally thought this graph was wrong, with more thought I realized that the only hypothesis that contains both 50 and 53 is that for all integers (1-100). Thus, each integer from 1-100 has to also be included in the same hypothesis, leading to each number having a probability plotted at 1.

In [47]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_e

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
```

```
    likelihood.append(like * prior)


for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.15384615384615383, 0.0, 0.7692307692307692, 0.0,
0.0, 0.0, 0.07692307692307691]
```



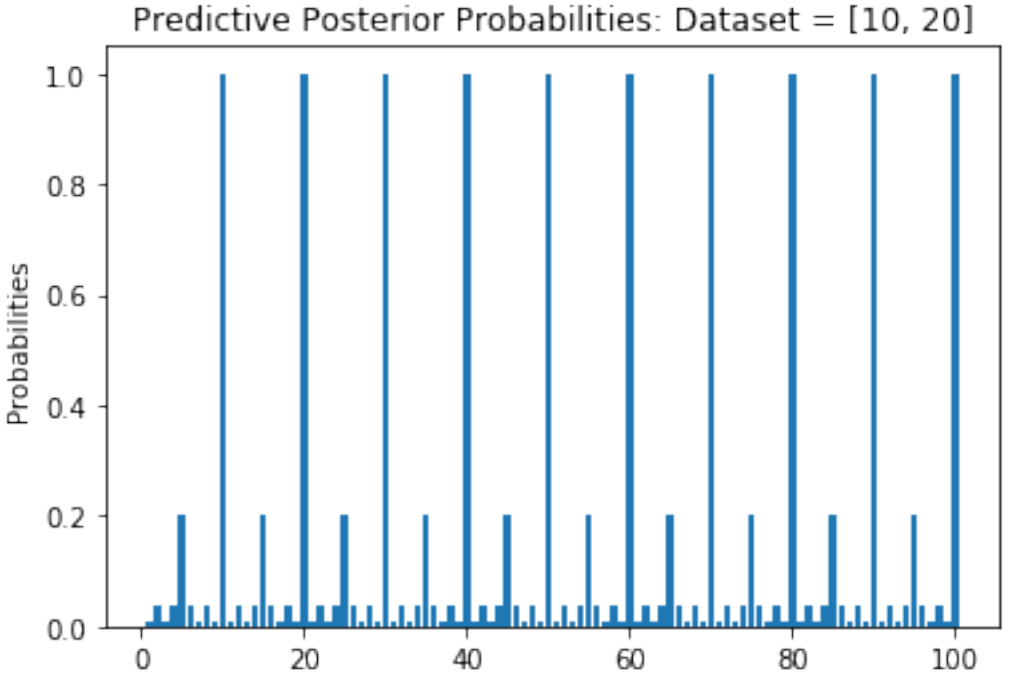
Explanation; Dataset = [16]: This matches my intuition that square numbers, especially even squares, are the most likely to also satisfy the hypothesis that accepts 16.

In [48]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_f

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
    likelihood.append(like * prior)
```

```
for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.030769230769230764, 0.0, 0.0, 0.0, 0.192307692307
69232, 0.7692307692307693, 0.0076923076923076911]
```



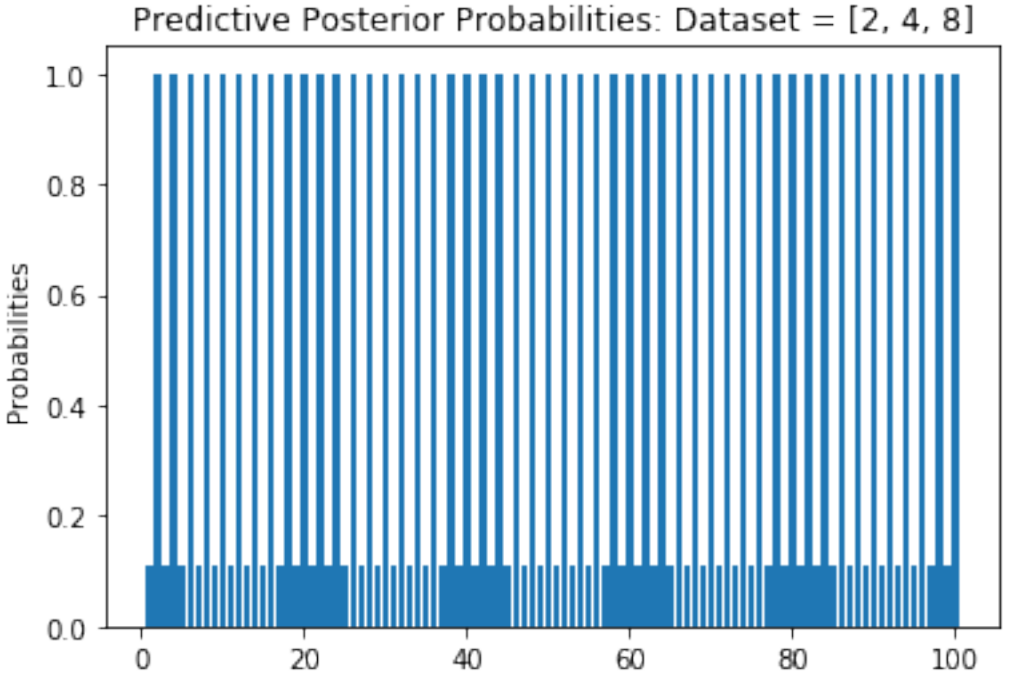
Explanation; Dataset = [10, 20]: This matches my intuition that numbers that are powers of 10 (and to some extent, powers of 5) would also satisfy the hypothesis that accepts [10,20].

In [49]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_g

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
```

```
    likelihood.append(like * prior)


for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities
print(likelihood)

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

```
[0.888888888888889, 0.0, 0.0, 0.0, 0.0, 0.0, 0.11111
111111111112]
```

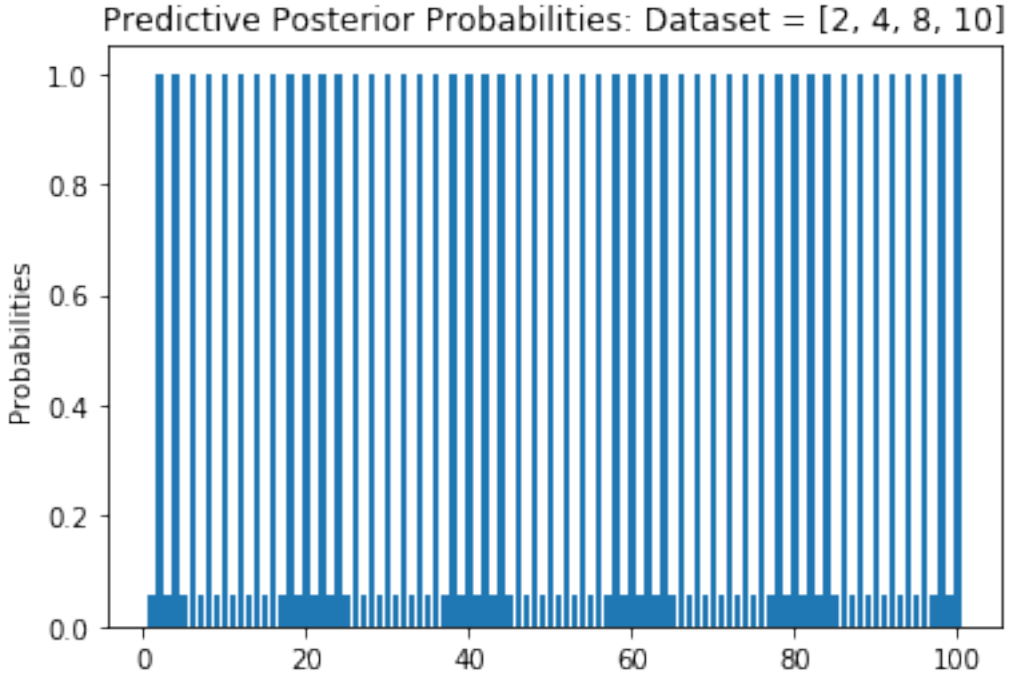


Explanation: Dataset = [2, 4, 8]: This matches my intuition that the most probable hypothesis is all even numbers, rather than all integers from (1, 100).

In [50]:

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = data_h

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

for h in hypotheses:

    like = splike(d, h)
    denom += like * prior
    likelihood.append(like * prior)
```

```
for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

# likelihood is now a list of predictive posterior probabilities

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



Explanation; Dataset = [2, 4, 8, 10]: This is in tune with my intuition that the hypothesis is probably all even numbers, rather than all integers from (1,100).

Question 3

In [51]:

```
#Calculating Hypotheses (ranges 1-100)
# h1 = even
# h2 = odd
# h3 = squares
# h4 = primes
# h5 = multiples of 5
# h6 = multiples of 10
# h7 = all nums

h1 = []
h2 = []
h3 = []
h4 = []
h5 = []
h6 = []
h7 = []

x = 1
i = 2

while x <= 100:
    if x%2 == 0:
        h1.append(x)
    else:
        h2.append(x)
    if x*x <= 100:
        h3.append(x*x)
    while i < x:
        if x%i == 0:
            break
        i += 1
    if i == x:
        h4.append(x)
    if x%5 == 0:
        h5.append(x)
    if x%10 == 0:
```

```
            h6.append(x)

        h7.append(x)
        x += 1
        i = 2

print("h1 (evens):", h1)
print("h2 (odds):", h2)
print("h3 (squares):", h3)
print("h4 (primes):", h4)
print("h5 (5x):", h5)
print("h6 (10x):", h6)
print("h7 (all):", h7)

# calculate the 8th hypothesis

h8 = []
start = 0
for start in range(101):
    end = start + 1
    for end in range(end, 101):
        h8.append([start, end])
```

```
h1 (evens): [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22,
24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48,
50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74,
76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100]
h2 (odds): [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 2
3, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 4
9, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 7
5, 77, 79, 81, 83, 85, 87, 89, 91, 93, 95, 97, 99]
h3 (squares): [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
h4 (primes): [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31
, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89
, 97]
h5 (5x): [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55,
60, 65, 70, 75, 80, 85, 90, 95, 100]
h6 (10x): [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
h7 (all): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26
, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39
, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52
, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65
, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78
, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91
, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```

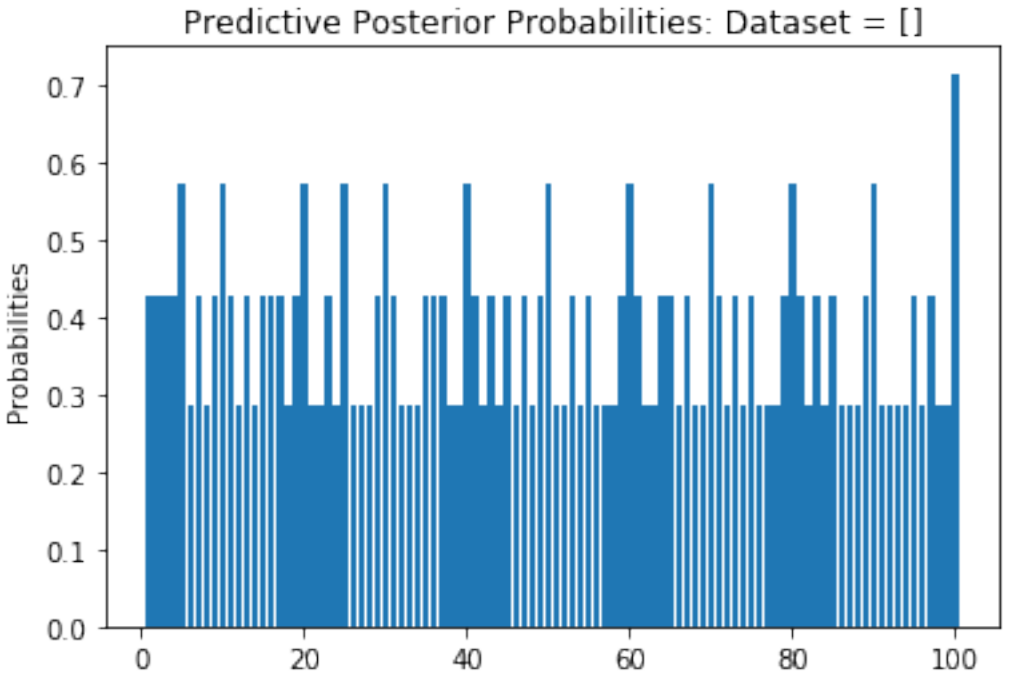
Question 3: There are 5,050 range-based hypotheses between 1-100.

In [57]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = []

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
```

```
prob_graph = []

prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
```

```
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



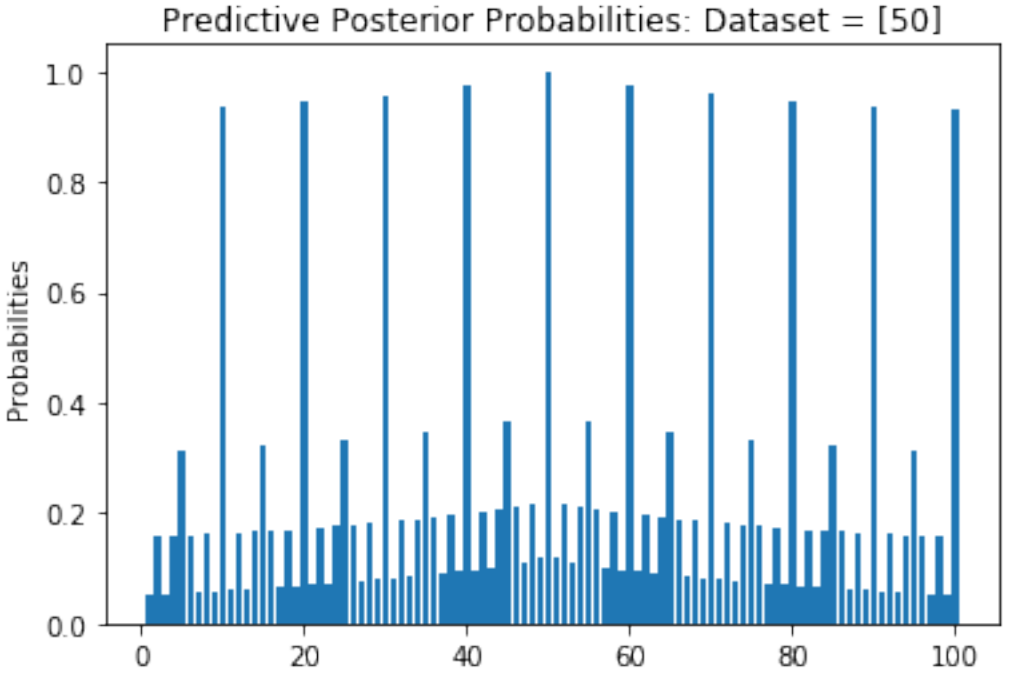
Explanation Dataset []: This graph has no data included, so the effects of the range-based hypotheses cannot be seen.

In [58]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = [50]

# define other needed variables
likelihood = []
like = 0
```

```
denom = 0

prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
```

```
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



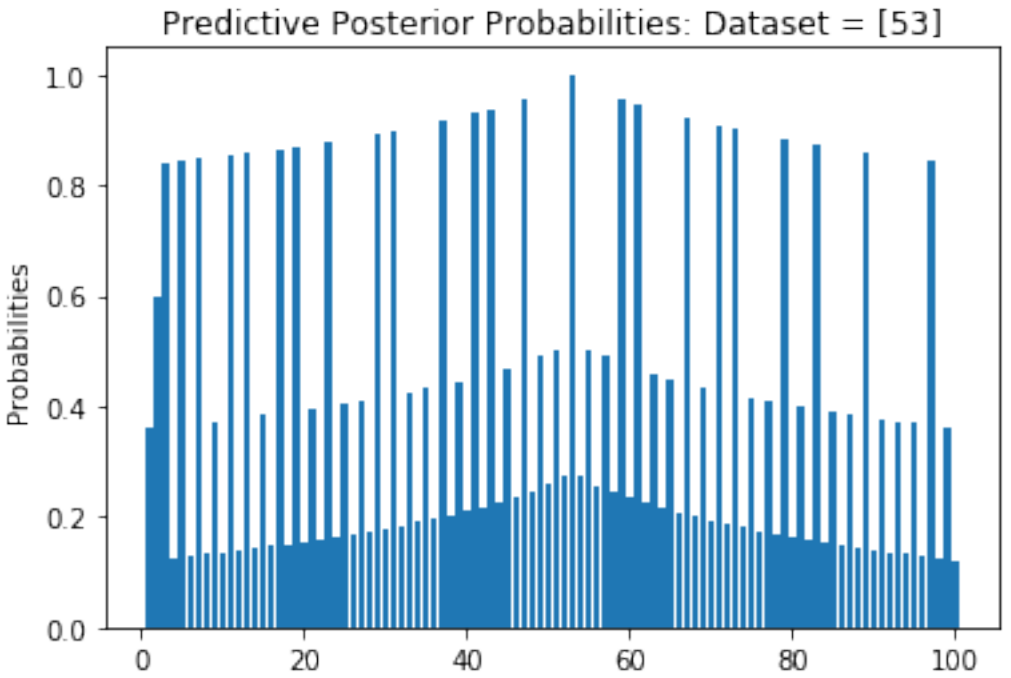
Explanation Dataset [50]: Here, as expected, the numbers which are multiples of 10 are expected with strong probability to be satisfied by the same hypothesis as 50, however the numbers closest to 50 have a slightly greater chance of also being satisfied by the same hypothesis because of their closeness in distance.

In [59]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = [53]
```

```
# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
```

```
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



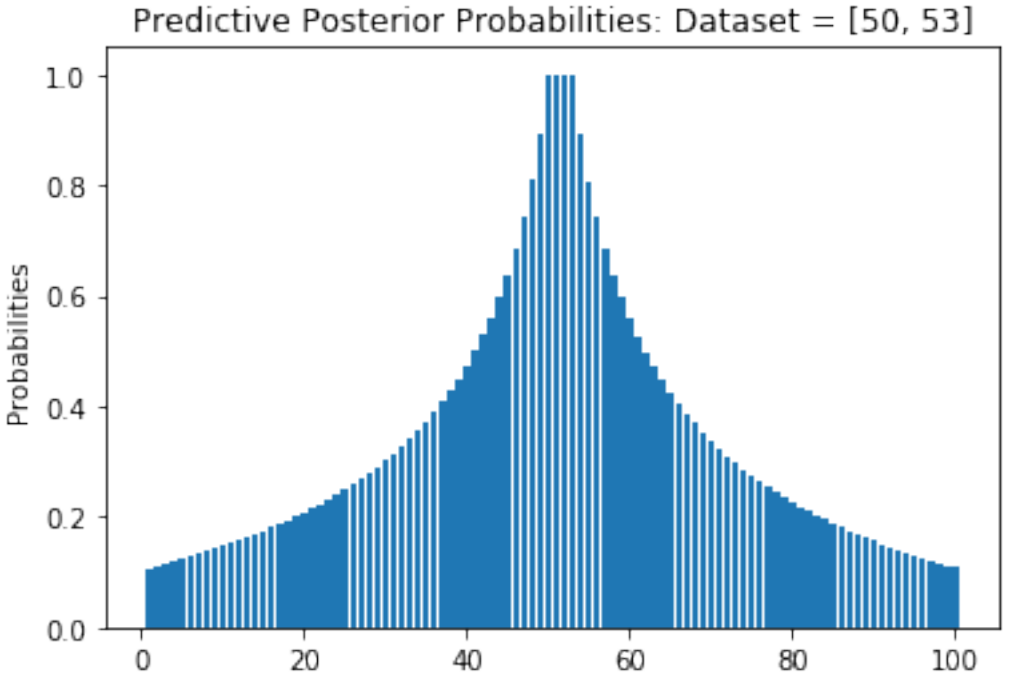
Explanation Dataset [53]: Here, as expected, the prime numbers are expected with strong probability to be satisfied by the same hypothesis as that which accepts [53], however the numbers closest to 53 have a slightly greater chance of also being satisfied by the same hypothesis because of their closeness in distance.

In [60]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
```

```
# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = [50, 53]

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
```

```
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```

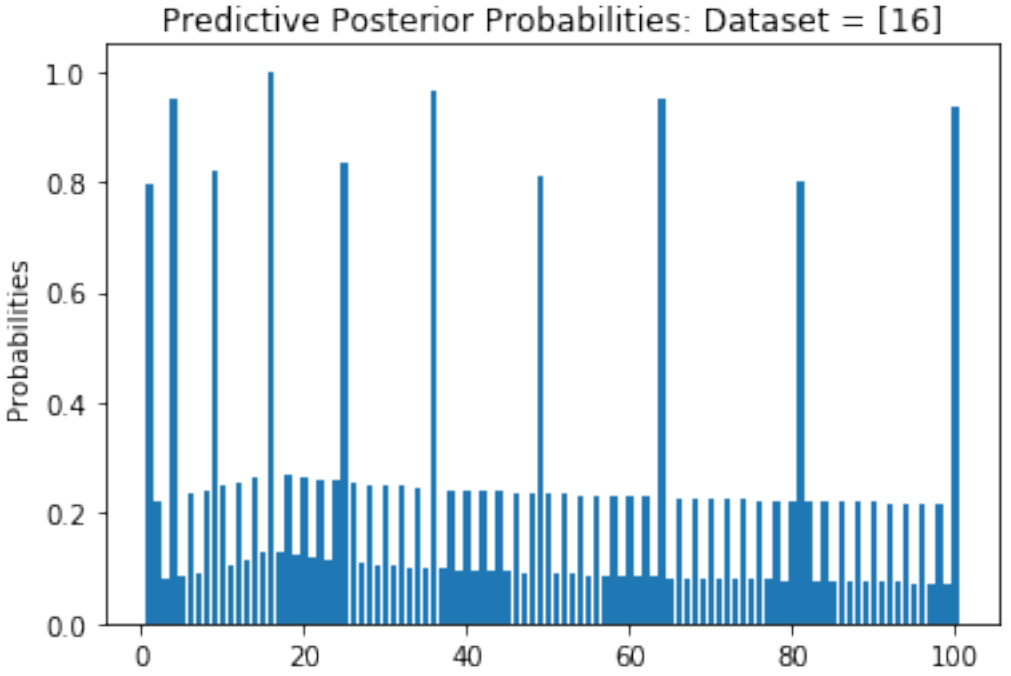


Explanation Dataset [50, 53]: Here is one of the graphs where my intuitions about the effects of the range-based hypotheses were strongest. In the previous question, the only hypothesis that satisfied 50 and 53 was that of all integers, so each number had an equal chance of satisfying the same hypothesis. However, now taking the range-based hypotheses into account, it can be seen that the probability of a number being satisfied by the same hypothesis is proportional to its distance away from the range in question, causing a symmetrical looking graph on the axis of 51.5 (directly in between 50 and 53).

In [61]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = [16]

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
```

```
for i in range(len(h8_relevant)):

    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



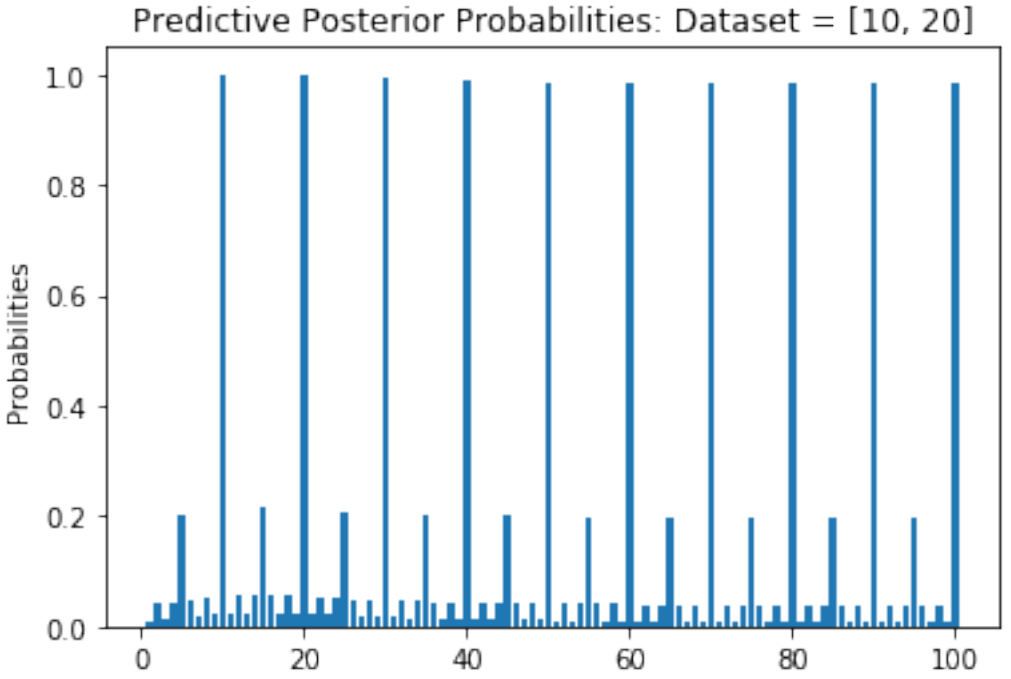
Explanation Dataset [16]: Here, as expected, the numbers which are squares (specifically even squares) are expected with strong probability to be satisfied by the same hypothesis as that which accepts [16], however the numbers closest to 16 have a slightly greater chance of also being satisfied by the same hypothesis because of their closeness in distance.

In [62]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = [10, 20]

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
```

```
# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()
```

```
# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



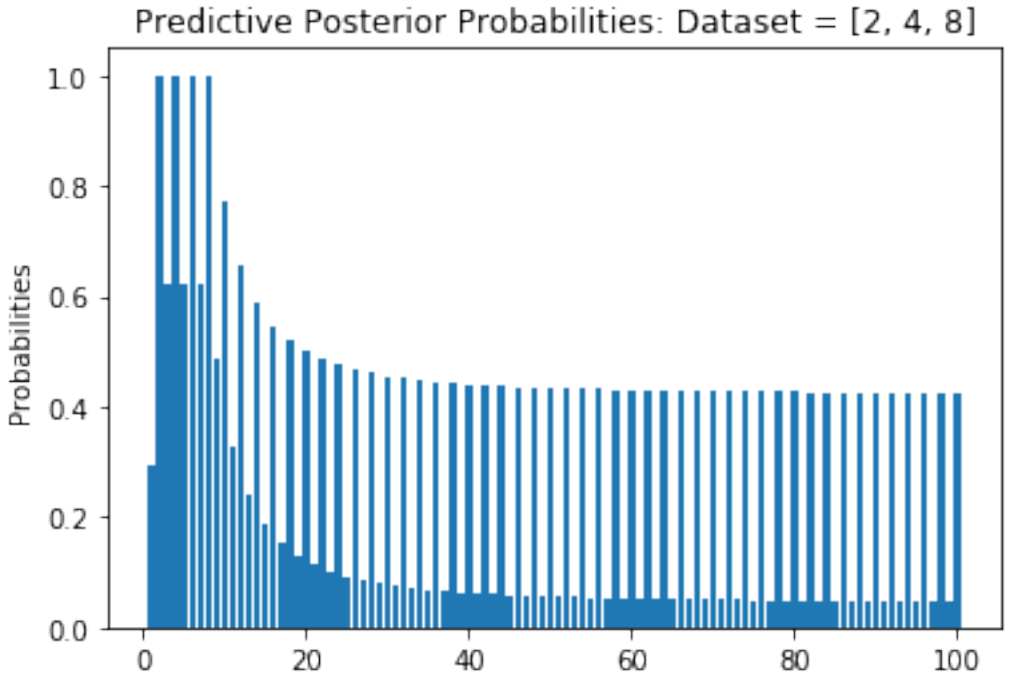
Explanation Dataset [10, 20]: Here, as expected, the numbers which are multiples of 10 are expected with strong probability to be satisfied by the same hypothesis as that which accepts 10 and 20, however the numbers closest to the (10, 20) interval have a slightly greater chance of also being satisfied by the same hypothesis because of their closeness in distance, as can be seen by a slight increase in probabilities in that part of the graph.

In [63]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
d = [2, 4, 8]

# define other needed variables
likelihood = []
```

```
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
```

```
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



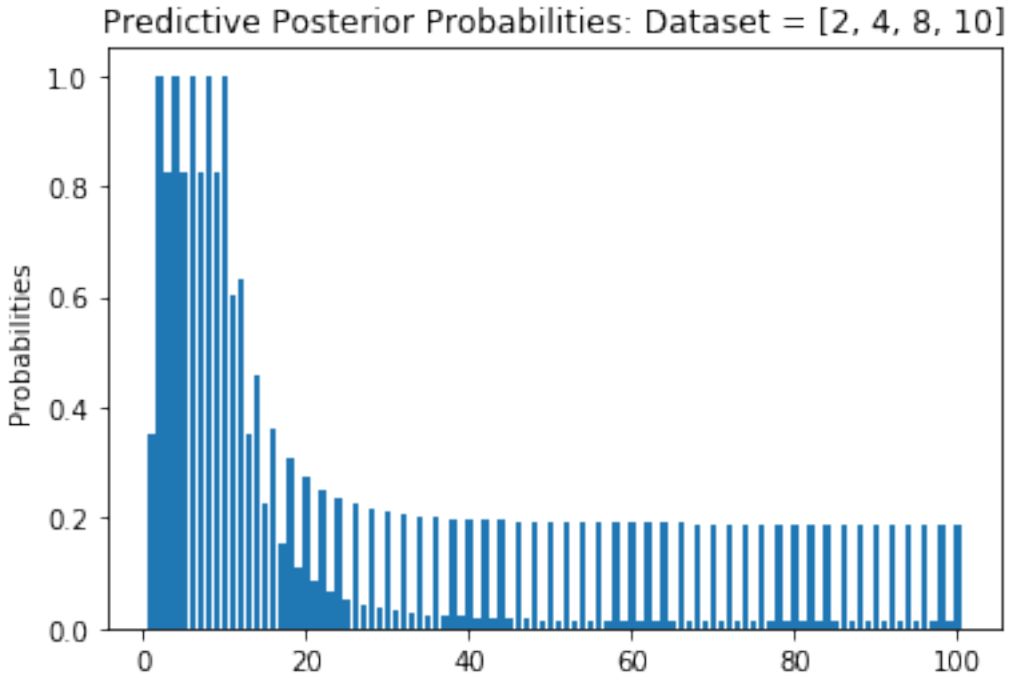
Explanation Dataset [2, 4, 8] As I expected, even numbers are still expected with stronger probability to be satisfied by hypothesis, relative to odd numbers. However, the even numbers closest to the (2, 8) interval have a slightly greater chance of also being satisfied by the same hypothesis because of their closeness in distance, as can be seen by a slight increase in probabilities in that part of the graph.

In [64]:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np

# define a list of hypotheses
hypotheses = [h1, h2, h3, h4, h5, h6, h7]

# determine which dataset we are using
```

```
# determine which dataset we are using
d = [2, 4, 8, 10]

# define other needed variables
likelihood = []
like = 0
denom = 0
prior = 1/len(hypotheses)
prob_graph = []
prob = 0

# find the relevant intervals which will be used for hypothesis
8
h8_relevant = []

for x in d:
    for i in range(len(h8)):
        if x >= h8[i][0] and x <= h8[i][1]:
            h8_relevant.append(h8[i])

# format each hypothesis in h8_relevant like the rest of the hyp
otheses
for i in range(len(h8_relevant)):
    h8_relevant[i] = list(range(h8_relevant[i][0] + 1, h8_releva
nt[i][1] + 1))

for h in h8_relevant:
    hypotheses.append(h)

for i in range(len(hypotheses)):
    if i < 7:
        prior = 1/8
    else:
        prior = ((1/8)/5050)
    like = splike(d, hypotheses[i])
    denom += like * prior
    likelihood.append(like * prior)

for i in range(len(likelihood)):
    likelihood[i] = likelihood[i]/denom

for i in range(1, 101):
    for c in range(len(hypotheses)):
        if i in hypotheses[c]:
            prob += likelihood[c]
```

```
            #print("here, hypothesis:", c)
    prob_graph.append(prob)
    prob = 0

title = "Predictive Posterior Probabilities: Dataset = " + str(d
)
plt.bar(np.arange(1, 101), prob_graph)
plt.ylabel('Probabilities')
plt.title(title)
plt.show()

# reset variables
denom = 0
prob = 0
prob_graph = []
likelihood = []
```



Explanation Dataset [2, 4, 8, 10]: Similar to the graph above, this graph shows that even numbers in the range from 2, 10 have the highest probability of satisfying the hypothesis, compared to other even numbers. However, a steeper drop-off in probability can be seen in this graph due to the fact that 2, 4, 8 and 10 are all so close to each other. More data that is close together makes for a stronger case that a range-based hypothesis is correct, making other hypotheses have a lower probability.