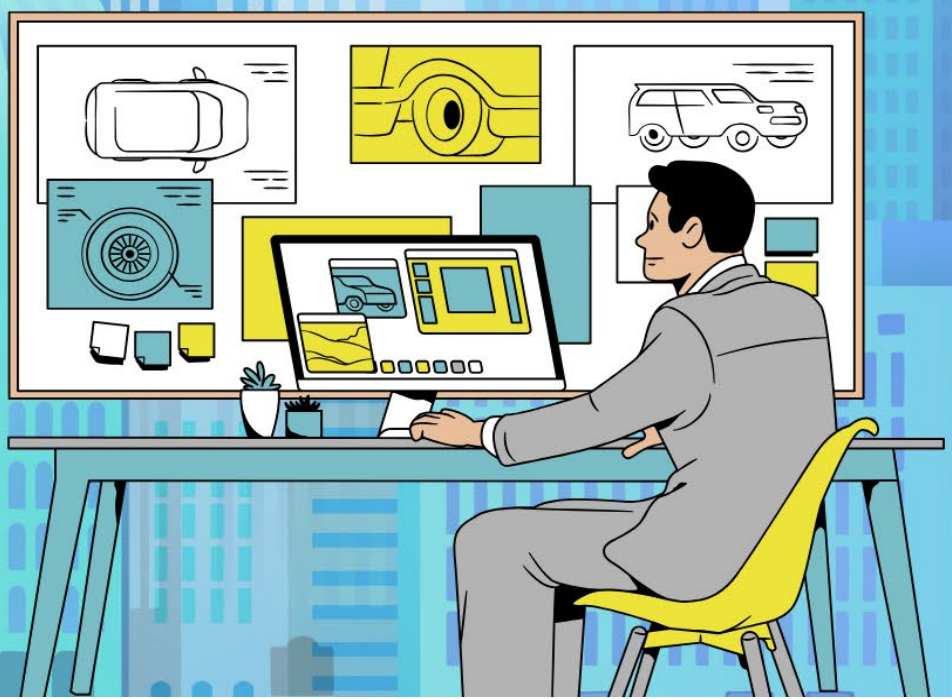
ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

ЗАДАЧНИК

Инженерный класс
в московской школе

НАПРАВЛЕНИЕ АВИАСТРОИТЕЛЬНЫЕ КЛАССЫ

ПРАКТИЧЕСКИЙ ЭТАП



МОСКВА
2025

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

ЗАДАЧНИК РАЗРАБОТАН:

Ткаченко Данилом Александровичем – Преподаватель Центра Детский технопарк МАИ «Траектория взлёта»

Бернацким Кириллом Сергеевичем – Ассистент кафедры 701 и Инженер НИО-701

Потешным Кириллом Александровичем - Инженер НИО-701 МАИ

Куяновым Владиславом Вячеславовичем - Инженер НИО-701 МАИ

МОСКВА
2025

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

СОДЕРЖАНИЕ

Вариант 1

Кейс №1 «Программирование»

«Диагональный анализ простых чисел»

Разработать на языке Python программу для анализа простых чисел, расположенных на главной и побочной диагоналях квадратной матрицы.

Пользователь вводит размер квадратной матрицы n, затем n × n целых положительных чисел. Требуется:
1. Сформировать матрицу n×n из введённых чисел;
2. Найти все простые числа на главной и побочной диагонали;
3. Вывести:
Список всех простых чисел на диагоналях, отсортированный по убыванию методом пузырьковой сортировки;

Сумму и среднее значение найденных простых чисел (с точностью до двух знаков после запятой);

Если число встречается на обеих диагоналях, оно включается в список один раз
4. Если на диагоналях нет простых чисел — вывести сообщение: «Простых чисел на диагоналях не найдено.»

Таблица с функциями:

Функция	**Параметры**	**Описание**	**Пример вызова функции**
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
bubble_sort	массив (список): **list[int]**	Сортировка массива методом пузырька по убыванию	bubble_sort([5, 3, 8])
find_diagonal_primes	двумерный массив(список): **list[list[int]]**	Нахождение простых чисел на главной и побочной диагоналях	find_diagonal_primes(matrix)
print_diagonal_info	список простых чисел: list[int]	Форматированный вывод результатов	print_diagonal_info(primes)

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размер матрицы n — от 1 до 100.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются)
Функция is_prime(n) проверяет простоту через деление до √n.
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
Если простых чисел на диагоналях нет, результатом является сообщение «Простых чисел на диагоналях не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:
def main():

```
n = int(input("Введите размер квадратной матрицы: "))

print("Введите элементы матрицы построчно:")
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные
Пример 1: Несколько простых чисел на обеих диагоналях

Ввод:

3

2 4 3

6 7 8

5 9 11

Вывод:

Простые числа на диагоналях (отсортированы по убыванию): [11, 7, 5, 3, 2]

Сумма: 28

Среднее значение: 5.60

Пример 2: Ни одного простого числа на диагоналях

Ввод:

3

4 6 8

9 10 12

14 15 16

Вывод:

Простых чисел на диагоналях не найдено.

Пример 3: Один простой элемент на пересечении диагоналей

Ввод:

1

13

Вывод:

Простые числа на диагоналях (отсортированы по убыванию): [13]

Сумма: 13

Среднее значение: 13.00

Пример 4: Простые только на побочной диагонали

Ввод:

3

4 6 5

9 7 10

11 12 8

Вывод:

Простые числа на диагоналях (отсортированы по убыванию): [11, 7, 5]

Сумма: 23

Среднее значение: 7.67

Пример 5: Все элементы на диагоналях — простые

Ввод:

2

2 3

5 7

Вывод:

Простые числа на диагоналях (отсортированы по убыванию): [7, 5, 3, 2]

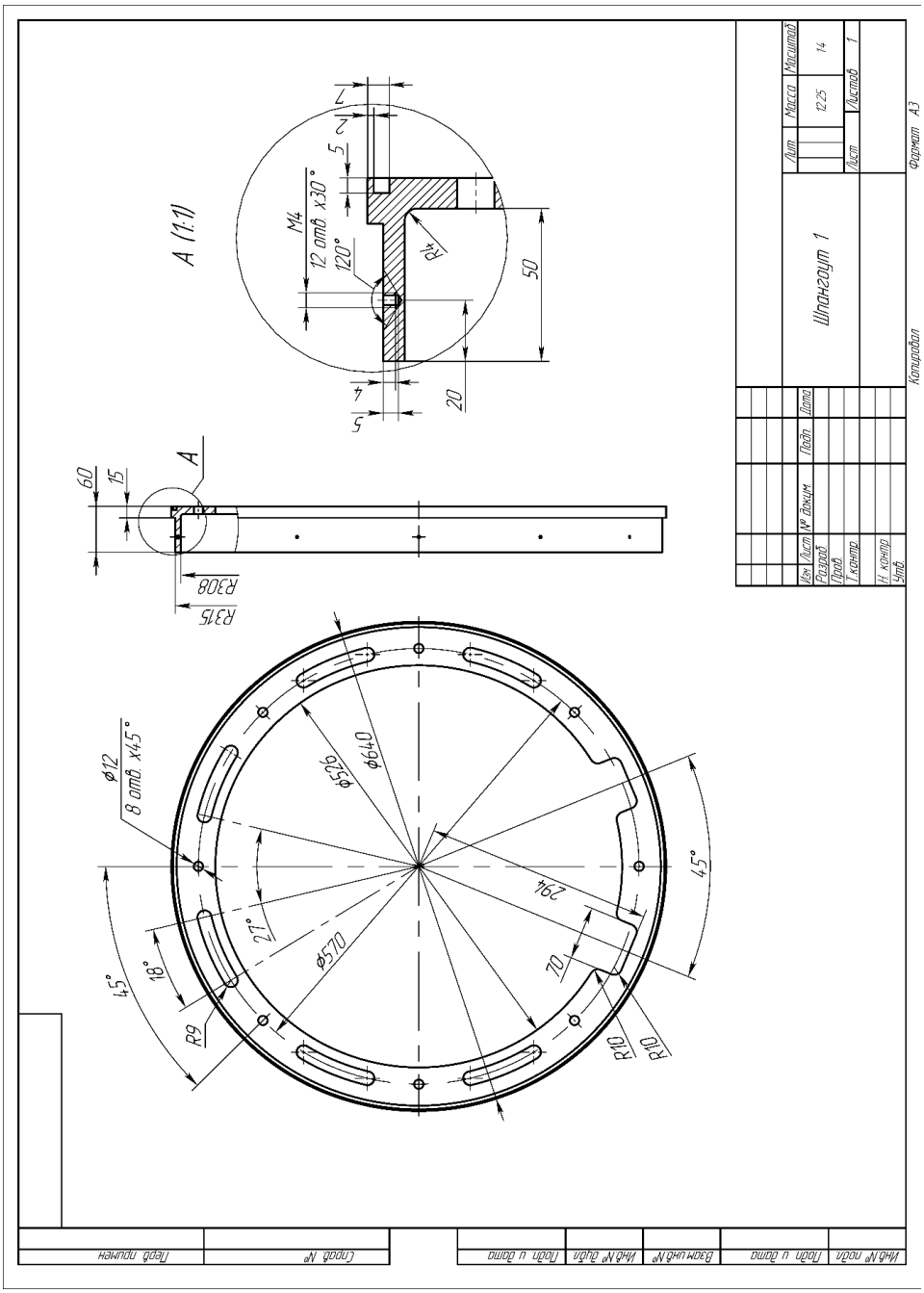
Сумма: 17

Среднее значение: 4.25

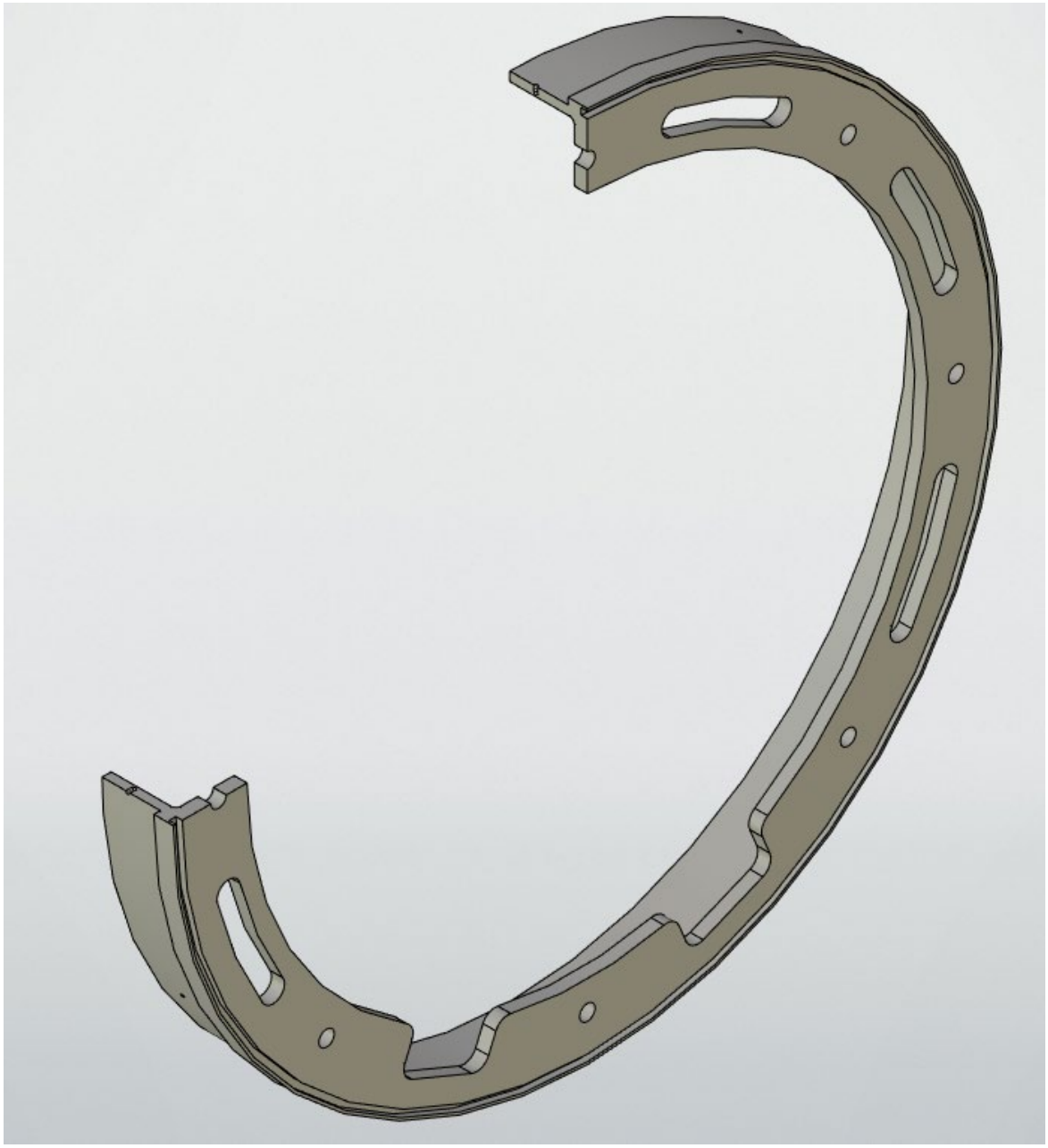
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.

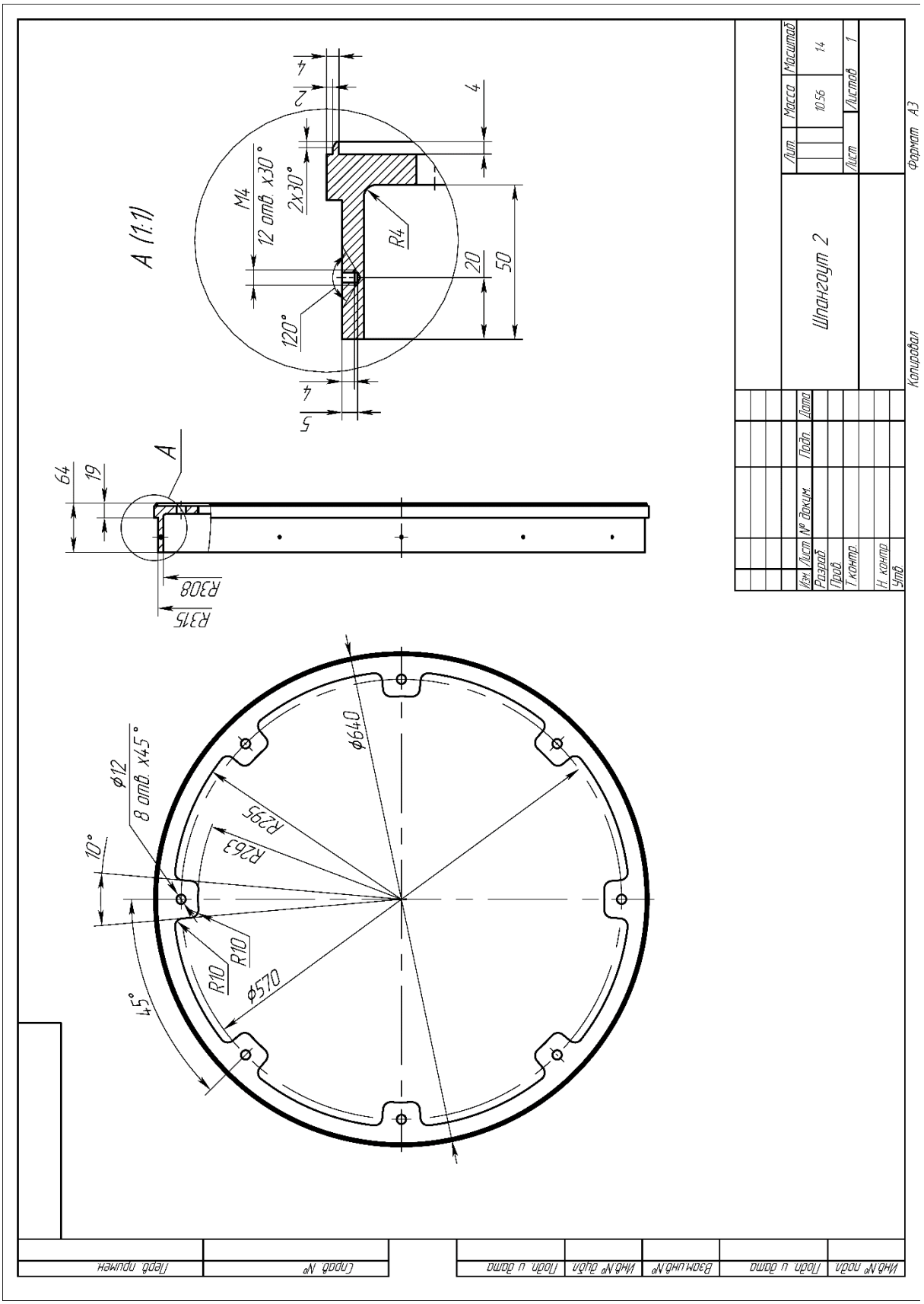


Ответ:

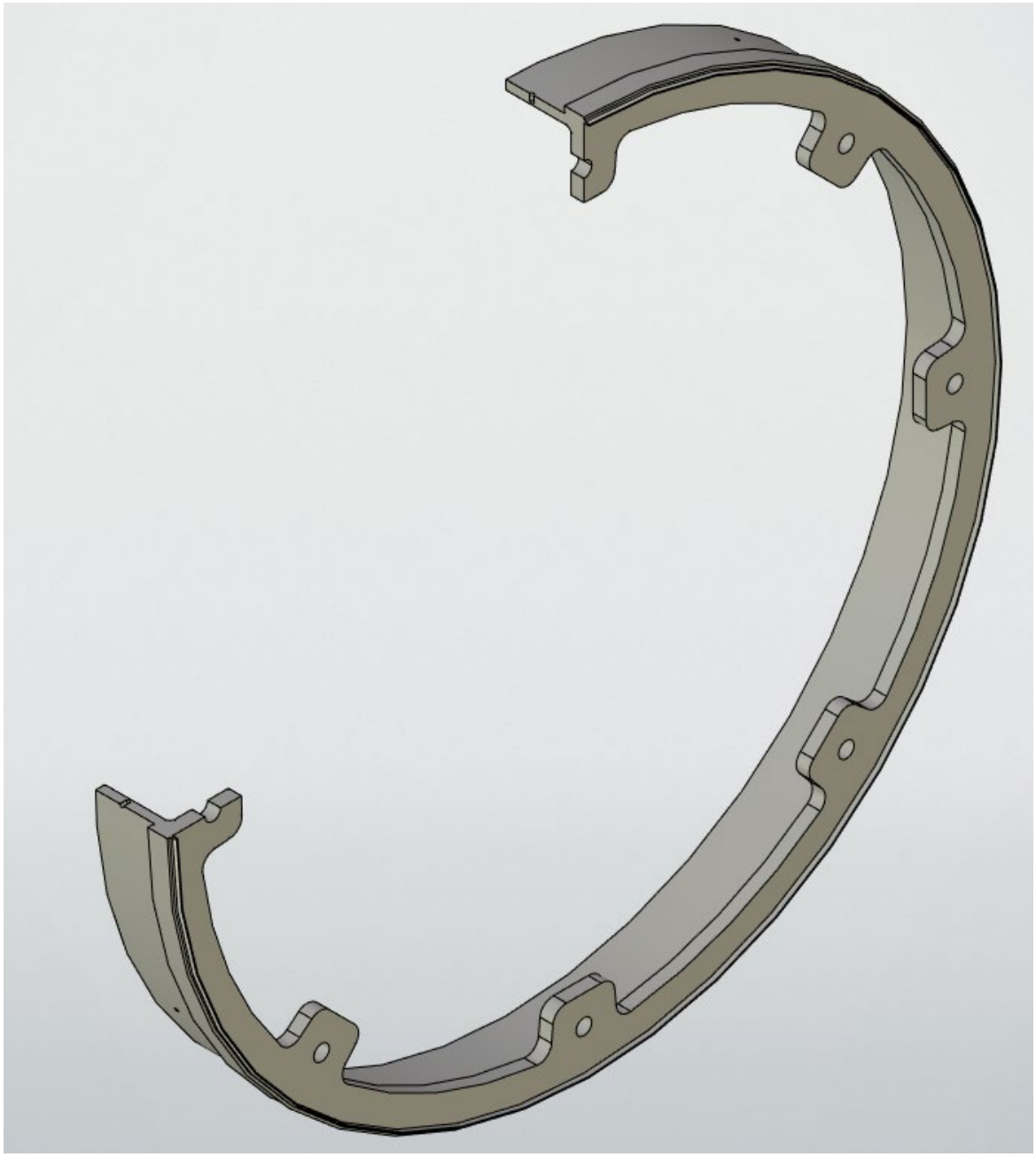


Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.

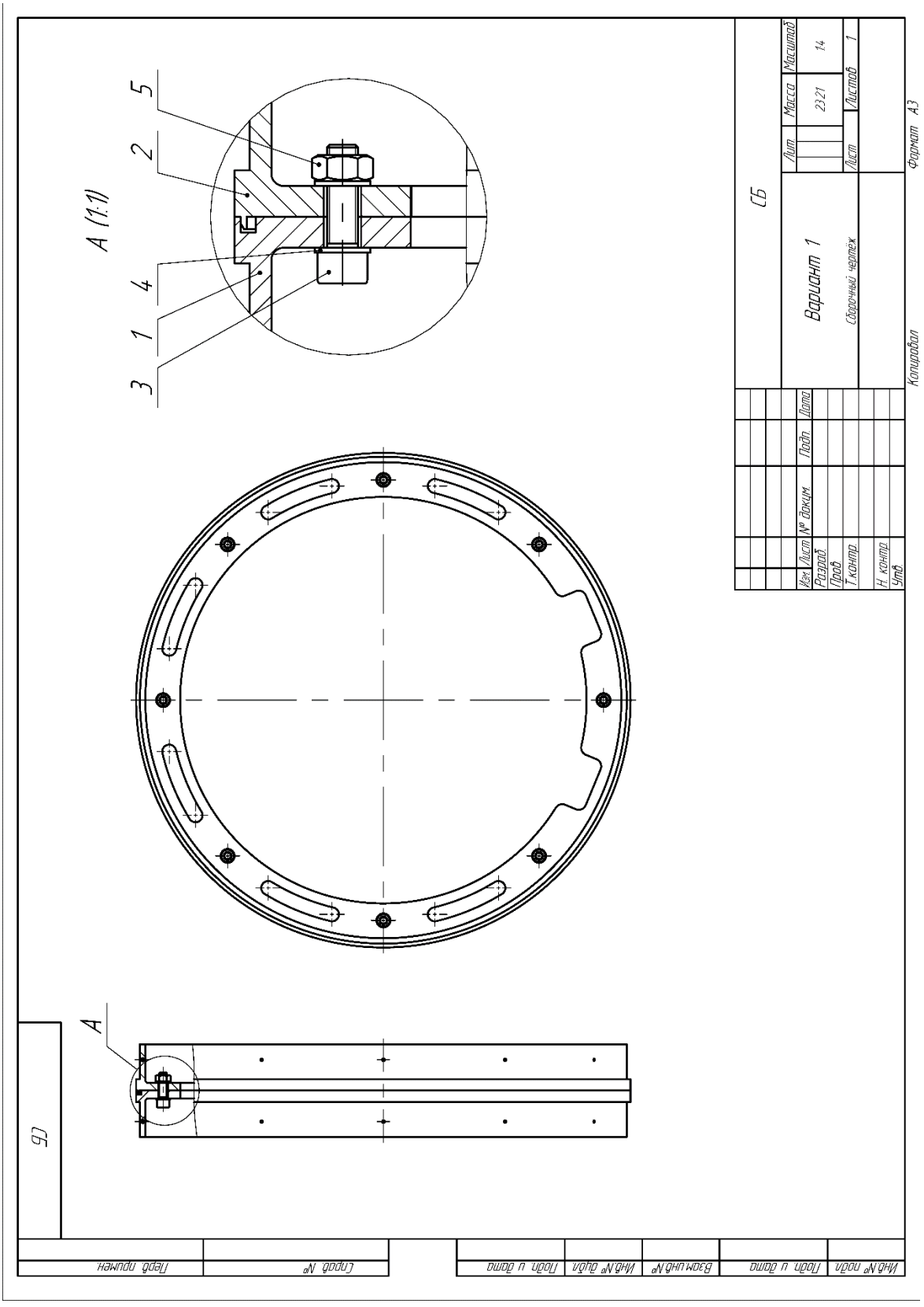


Ответ:

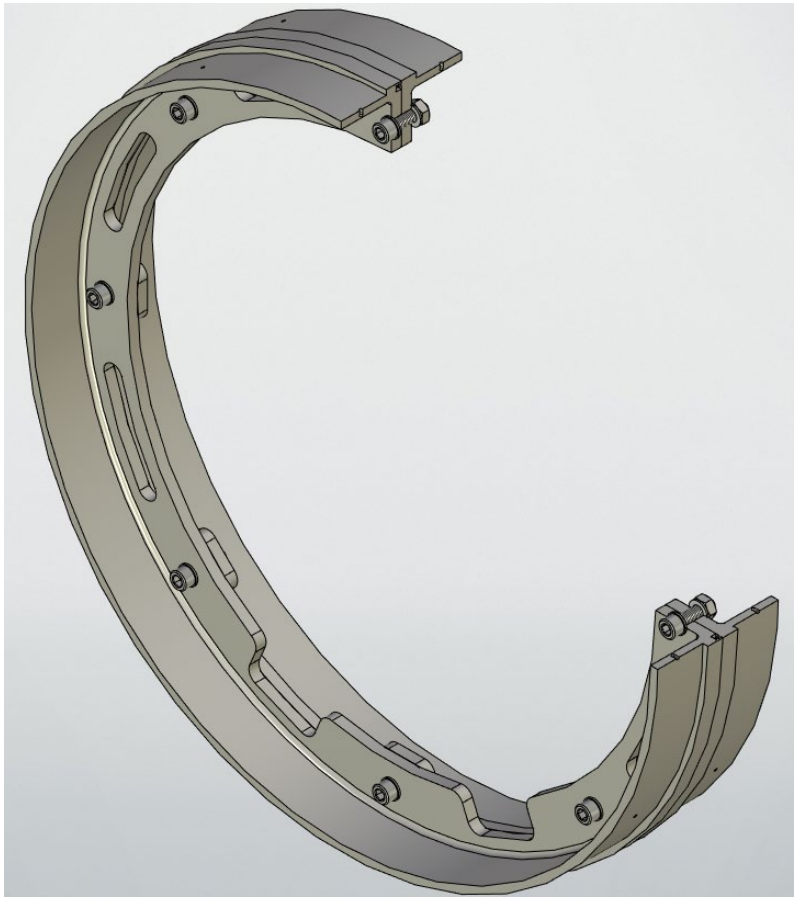


Задание №3

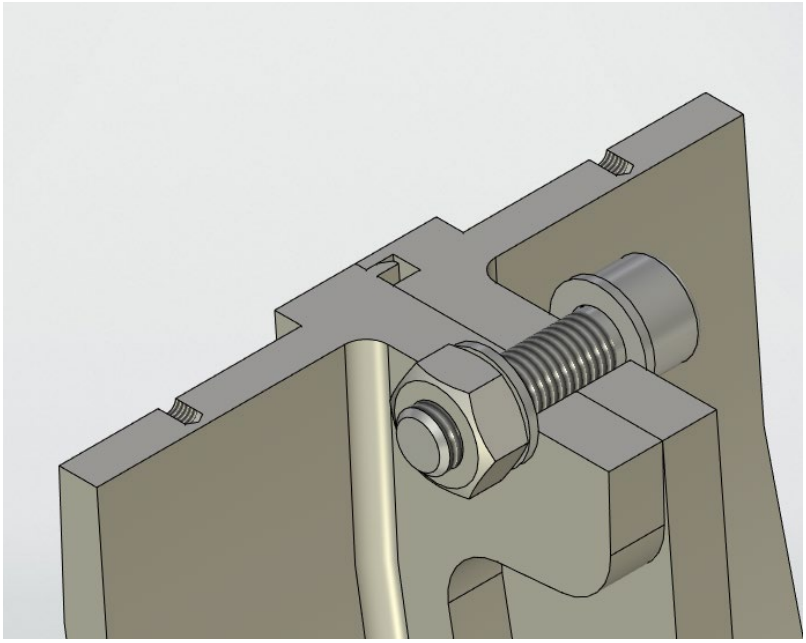
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Шпангоут 1	1	
А3		2		Шпангоут 2	1	
				Стандартные изделия		
		3		Винт М10×35		
				ГОСТ 11738-84	8	
		4		Гайка М10		
				ГОСТ 5927-70	8	
		5		Шайба А 10		
				ГОСТ 10450-78	16	

Перв. примен.
Справ. №
Подп. и дата
Инв.№ дубл.
Взам.инв.№
Подп. и дата
Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 1

Лит.	Лист	Листов
		1

Копировал

Формат А4

Вариант 2

Кейс №1 «Программирование»

«Фильтр нечётных простых на границах»

Разработать на языке Python программу, которая находит все нечётные простые числа, расположенные на границах матрицы, и обрабатывает их.

Пользователь вводит размеры матрицы n и m, затем n × m целых положительных чисел. Требуется:

Сформировать матрицу n × m из введённых чисел;

Найти все нечётные простые числа, расположенные:

- в первой и последней строках;
- в первом и последнем столбцах;

Собрать нечётные простые числа в один список без повторений;

Отсортировать список методом пузырьковой сортировки по возрастанию;

Вывести:

- Список найденных чисел;
- Их сумму;
- Среднее значение (округлённое до двух знаков);

Если таких чисел нет — вывести сообщение: «Нечётных простых чисел на границах не найдено.»

Таблица с функциями:

Функция	**Параметры**	**Описание**	**Пример вызова функции**
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
bubble_sort	массив (список): **list[int]**	Сортировка пузырьком по возрастанию	bubble_sort([5, 3, 8])
find_border_primes	двумерный массив(список) : **list[list[int]]**	Поиск нечётных простых на границах	find_border_primes(matrix)

print_border_info	список простых чисел: list[int]	Форматированный вывод результатов	print_border_info(primes)

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размеры n и m от 1 до 100.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются)
Функция is_prime(n) проверяет простоту через деление до √n.
Простые числа определяются вручную, без sympy и других библиотек.
Повторы исключаются (одинаковое число на двух границах учитывается один раз).
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
При отсутствии подходящих чисел выводится сообщение: «Нечётных простых чисел на границах не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:
def main():

```
n, m = map(int, input("Введите размеры матрицы: ").split())

print("Введите элементы матрицы построчно:")
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные

Пример 1: Несколько нечётных простых на границах

Ввод:

3 4

3 4 5 6

8 9 10 11

13 14 15 17

Вывод:

Нечётные простые на границах (отсортированы): [3, 5, 11, 13, 17]

Сумма: 49

Среднее значение: 9.80

Пример 2: Только одно подходящее число в углу

Ввод:

2 2

4 2

6 7

Вывод:

Нечётные простые на границах (отсортированы): [7]

Сумма: 7

Среднее значение: 7.00

Пример 3: Повторяющееся число на границах — учитывается один раз

Ввод:

3 3

3 4 3

5 6 7

3 8 3

Вывод:

Нечётные простые на границах (отсортированы): [3, 5, 7]

Сумма: 15

Среднее значение: 5.00

Пример 4: Все границы есть, но ни одно число не подходит

Ввод:

3 3

2 4 6

8 10 12

14 16 18

Вывод:

Нечётных простых чисел на границах не найдено.

Пример 5: Все простые на одной стороне

Ввод:

2 5

3 5 7 11 13

4 6 8 10 12

Вывод:

Нечётные простые на границах (отсортированы): [3, 5, 7, 11, 13]

Сумма: 39

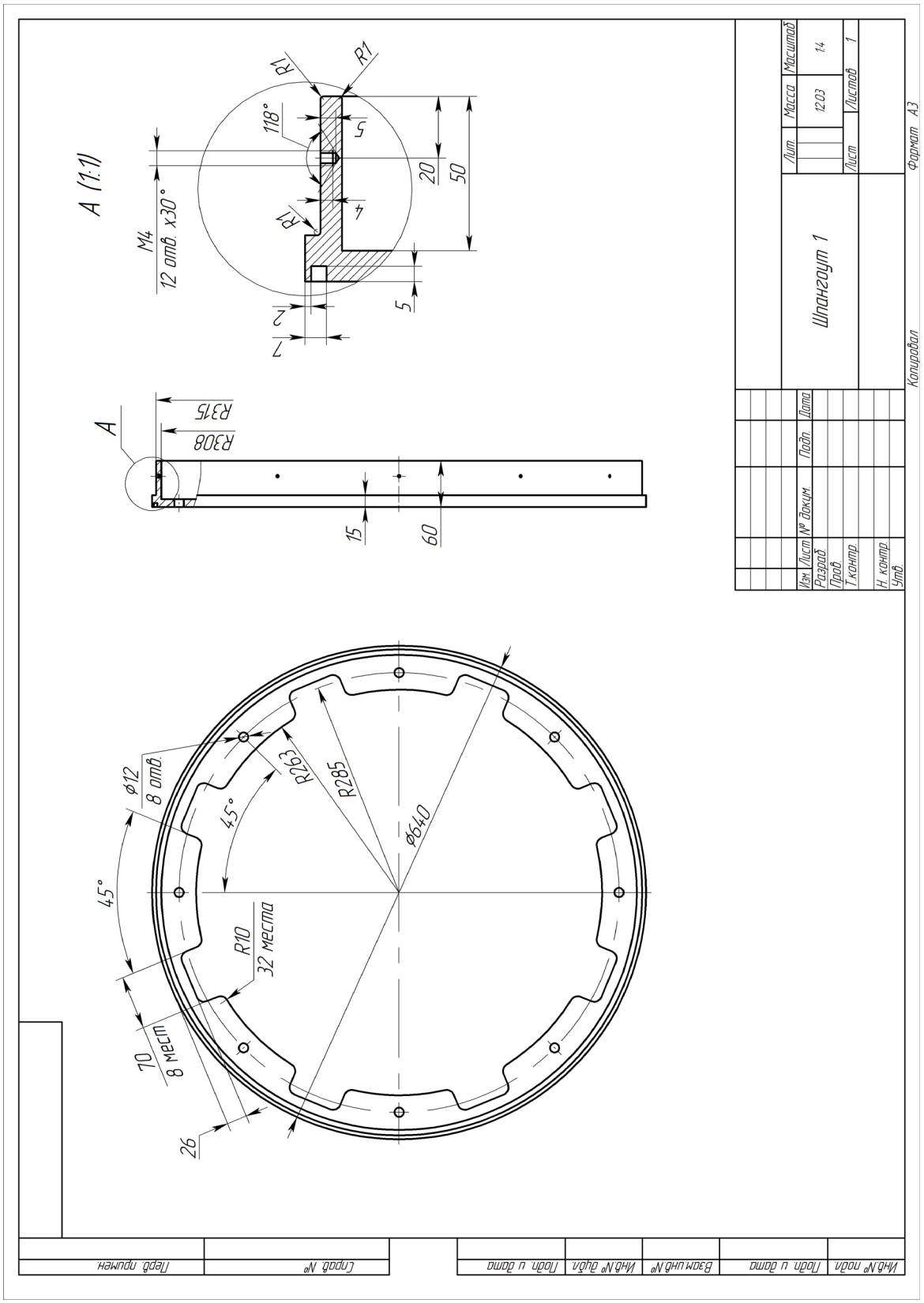
Среднее значение: 7.80

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

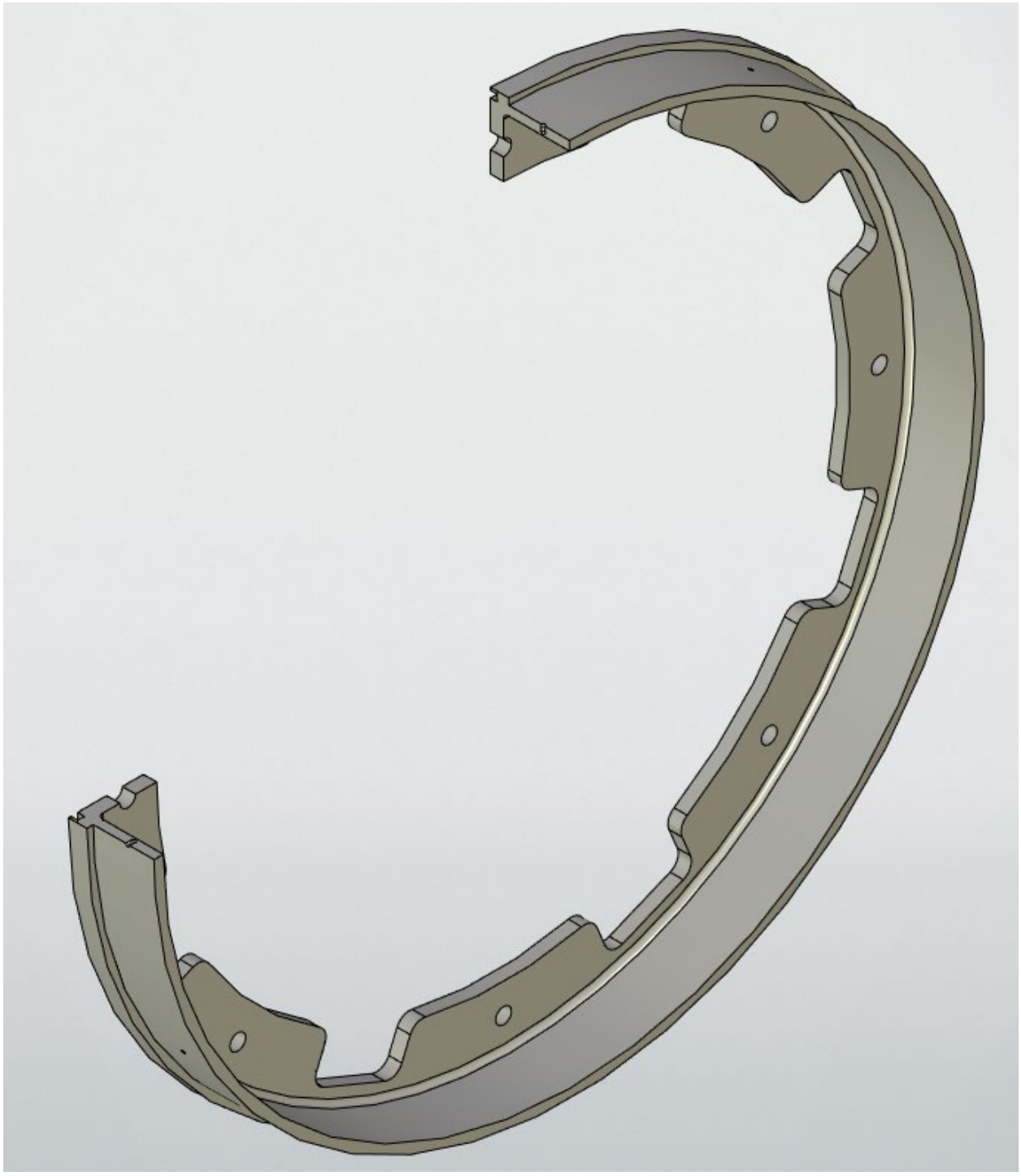
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



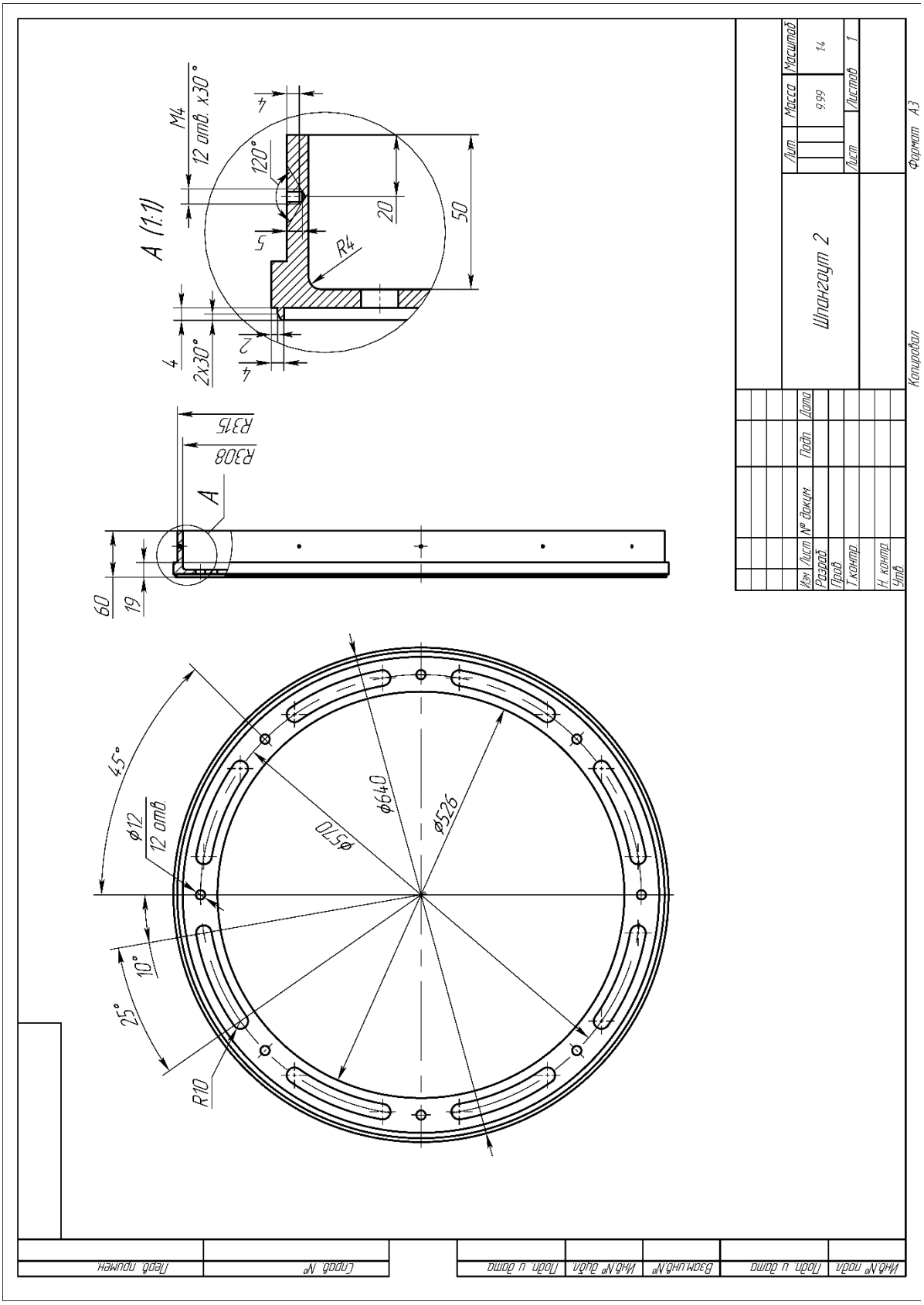
Ответ:



Рассечённая модель шпангоута в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.

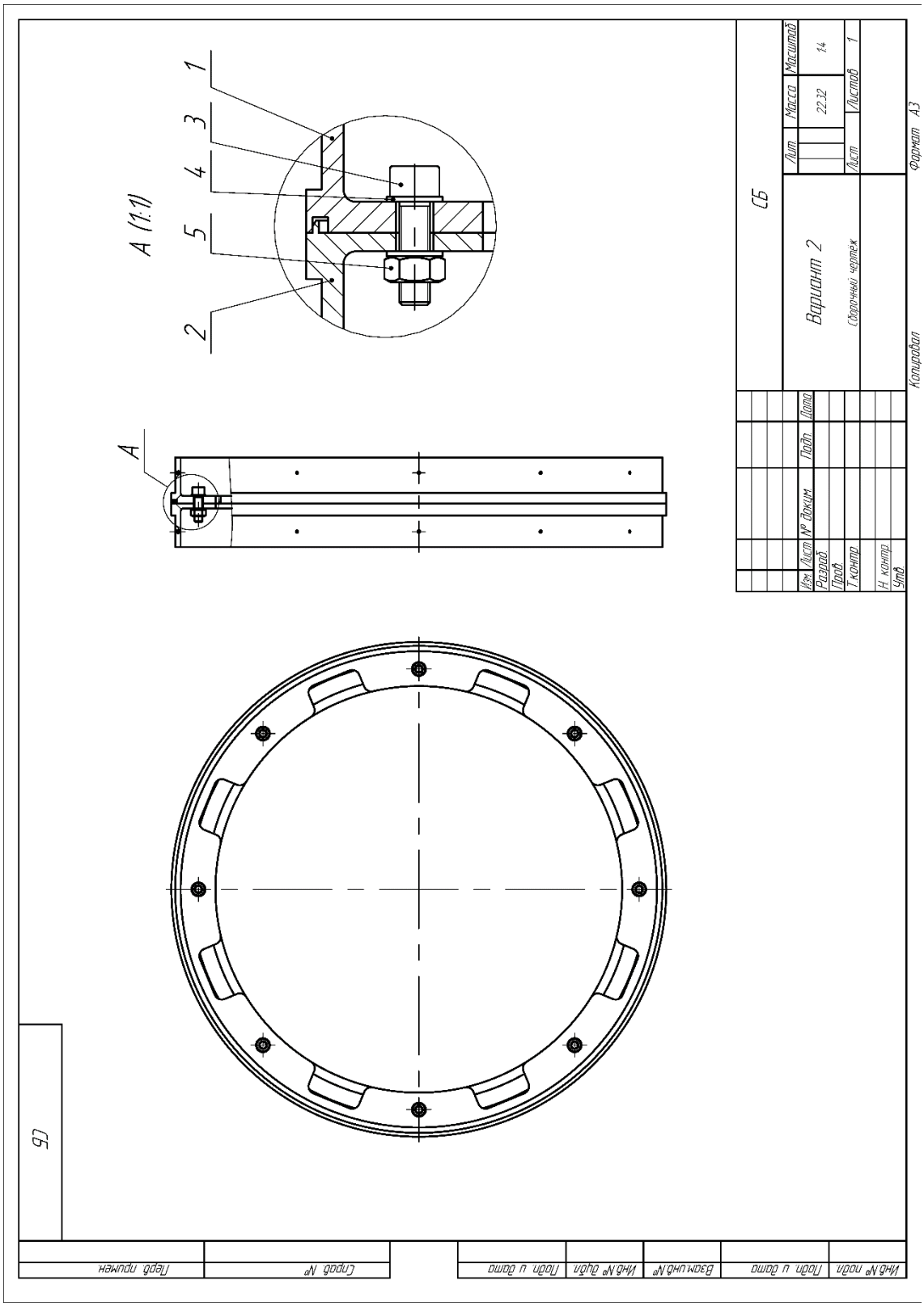


Ответ:

Рассечённая модель шпангоута в изометрии

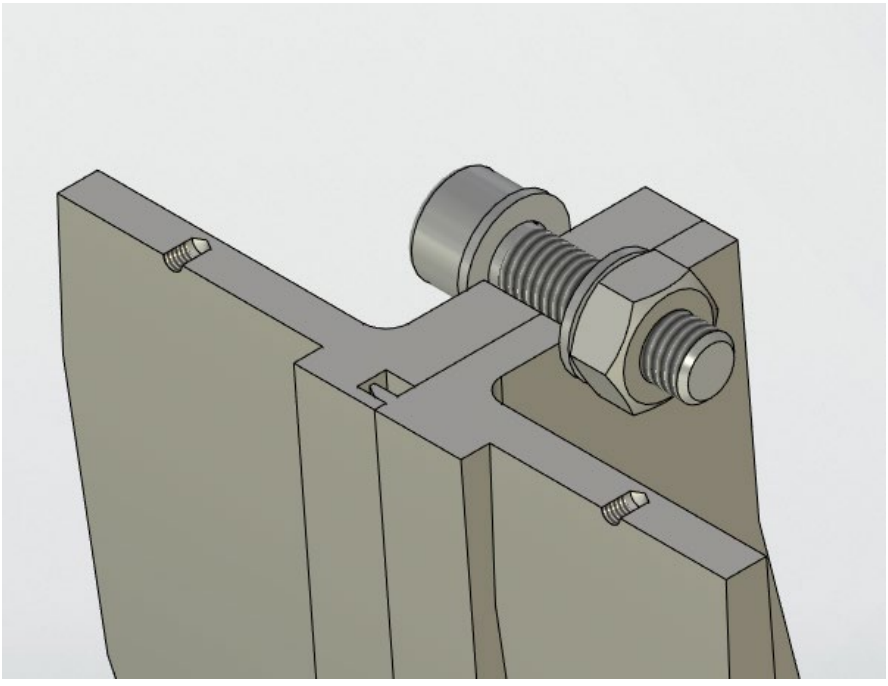
Задание №3

Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:

Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Шпангоут 1	1	
А3		2		Шпангоут 2	1	
				Стандартные изделия		
		3		Винт М10×35		
				ГОСТ 11738-84	8	
		4		Гайка М10		
				ГОСТ 5927-70	8	
		5		Шайба А 10		
				ГОСТ 10450-78	16	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 2

Лит.	Лист	Листов
		1

Копировал

Формат А4

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

Вариант 3

Кейс №1 «Программирование»

«Простые числа в диапазоне»

Разработать на языке Python программу, которая находит все простые числа в матрице, попадающие в заданный пользователем диапазон [a, b], и анализирует их.

Пользователь вводит размеры матрицы n и m, затем n × m целых положительных чисел. После этого вводит два числа a и b, задающих диапазон (включительно). Если a> b, программа должна считать диапазон [b, a]. Требуется:

Сформировать матрицу n × m из введённых чисел;

Найти все простые числа, находящиеся в диапазоне от a до b включительно;

Сортировать найденные простые числа методом пузырьковой сортировки по возрастанию;

Вывести:

- Отсортированный список простых чисел;
- Сумму найденных чисел;
- Среднее значение (округлённое до двух знаков);

Если подходящих простых чисел нет — вывести сообщение: «Простых чисел в диапазоне не найдено.»

Таблица с функциями:

Функция	**Параметры**	**Описание**	**Пример вызова функции**
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
bubble_sort	массив (список): **list[int]**	Сортировка пузырьком по возрастанию	bubble_sort([5, 3, 8])
find_primes_in_range	matrix: list[list[int]], a: int, b: int	Поиск всех простых чисел в диапазоне [a, b]	find_primes_in_range(matrix, 5, 17)
print_range_info	primes: list[int]	Форматированный вывод результатов	print_range_info(primes)

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размеры n и m от 1 до 100.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются)
Функция is_prime(n) проверяет простоту через деление до √n.
Простые числа определяются вручную, без sympy и других библиотек.
Диапазон [a, b] может быть введён в любом порядке — числа нужно отсортировать внутри кода.
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
Допускается, что a == b.
При отсутствии подходящих чисел выводится сообщение: «Простых чисел в диапазоне не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:

```
def main():

    n, m = map(int, input("Введите размеры матрицы: ").split())

    print("Введите элементы матрицы построчно:")

    ... #ввод элементов

    a, b = map(int, input("Введите границы диапазона a и b: ").split())
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные

Пример 1: Простые в диапазоне от 5 до 13

Ввод:

3 3

2 4 5

6 11 13

17 19 23

5 13

Вывод:

Простые числа в диапазоне (отсортированы): [5, 11, 13]

Сумма: 29

Среднее значение: 9.67

Пример 2: Диапазон наоборот (b < a)

Ввод:

2 2

2 3

4 5

10 2

Вывод:

Простые числа в диапазоне (отсортированы): [2, 3, 5]

Сумма: 10

Среднее значение: 3.33

Пример 3: Нет подходящих чисел

Ввод:

2 3

2 3 5

7 11 13

20 30

Вывод:

Простых чисел в диапазоне не найдено.

Пример 4: Только одно число в диапазоне

Ввод:

3 3

1 2 3

4 5 6

7 8 9

5 5

Вывод:

Простые числа в диапазоне (отсортированы): [5]

Сумма: 5

Среднее значение: 5.00

Пример 5: Все числа в диапазоне — составные

Ввод:

2 2

4 6

8 10

4 10

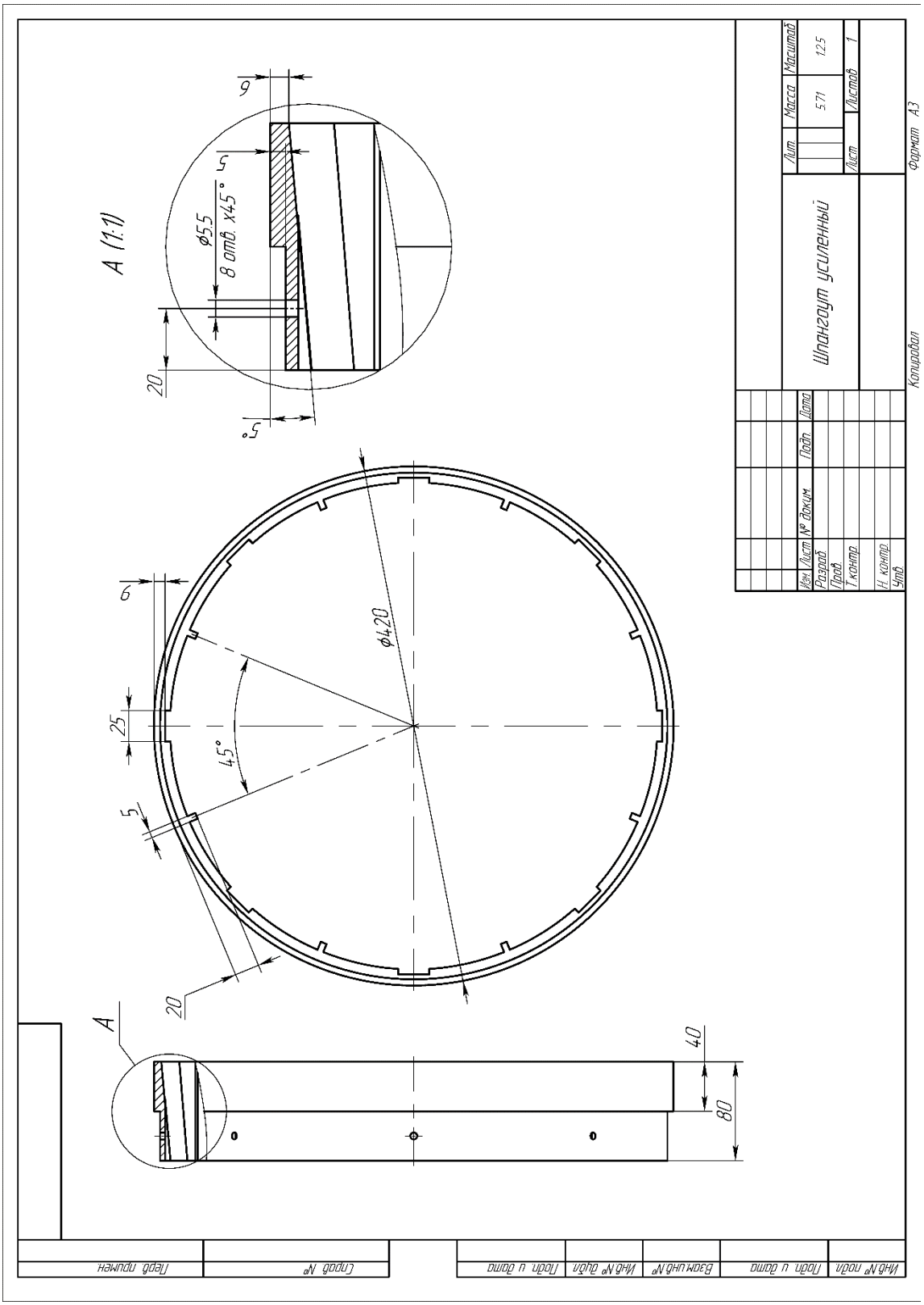
Вывод:

Простых чисел в диапазоне не найдено.

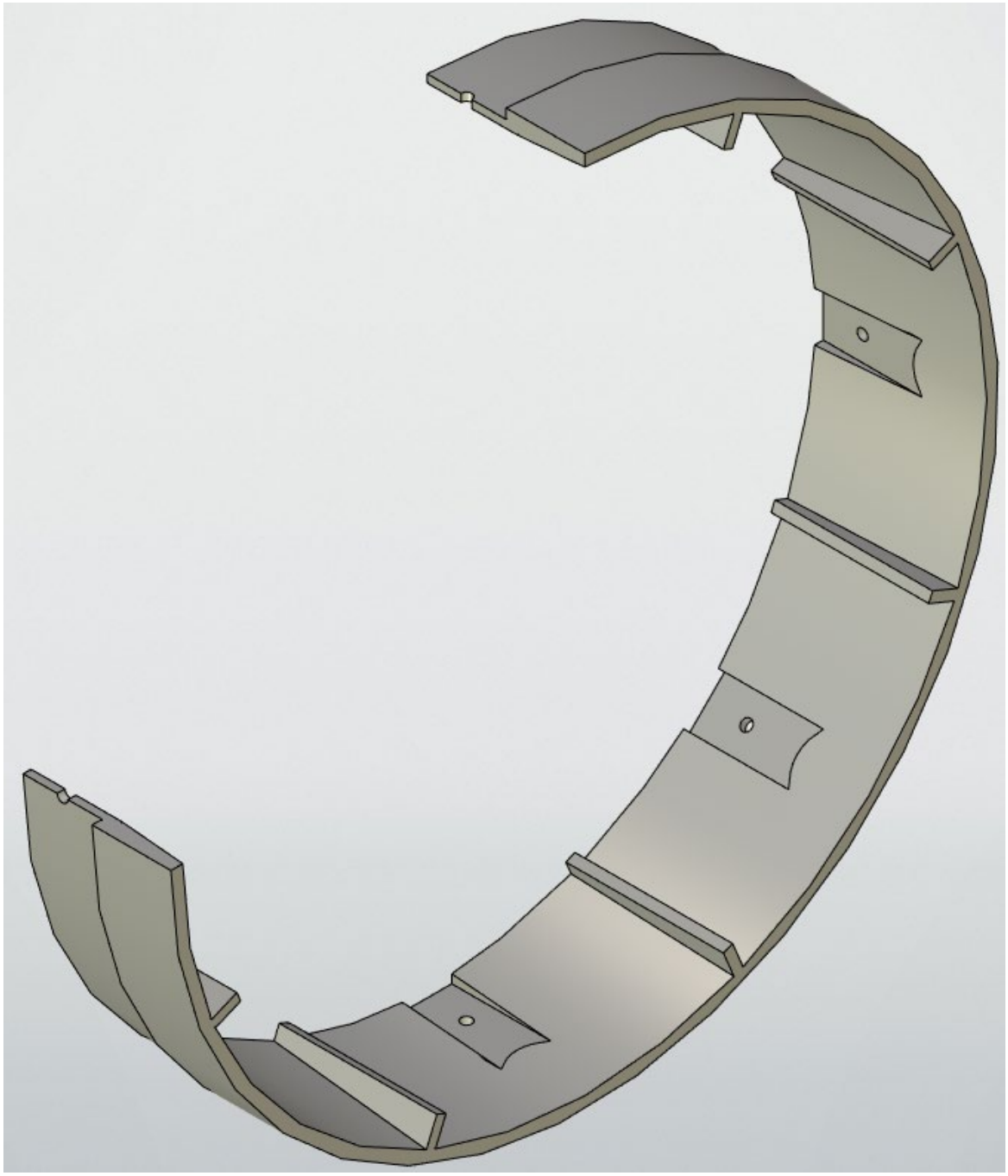
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



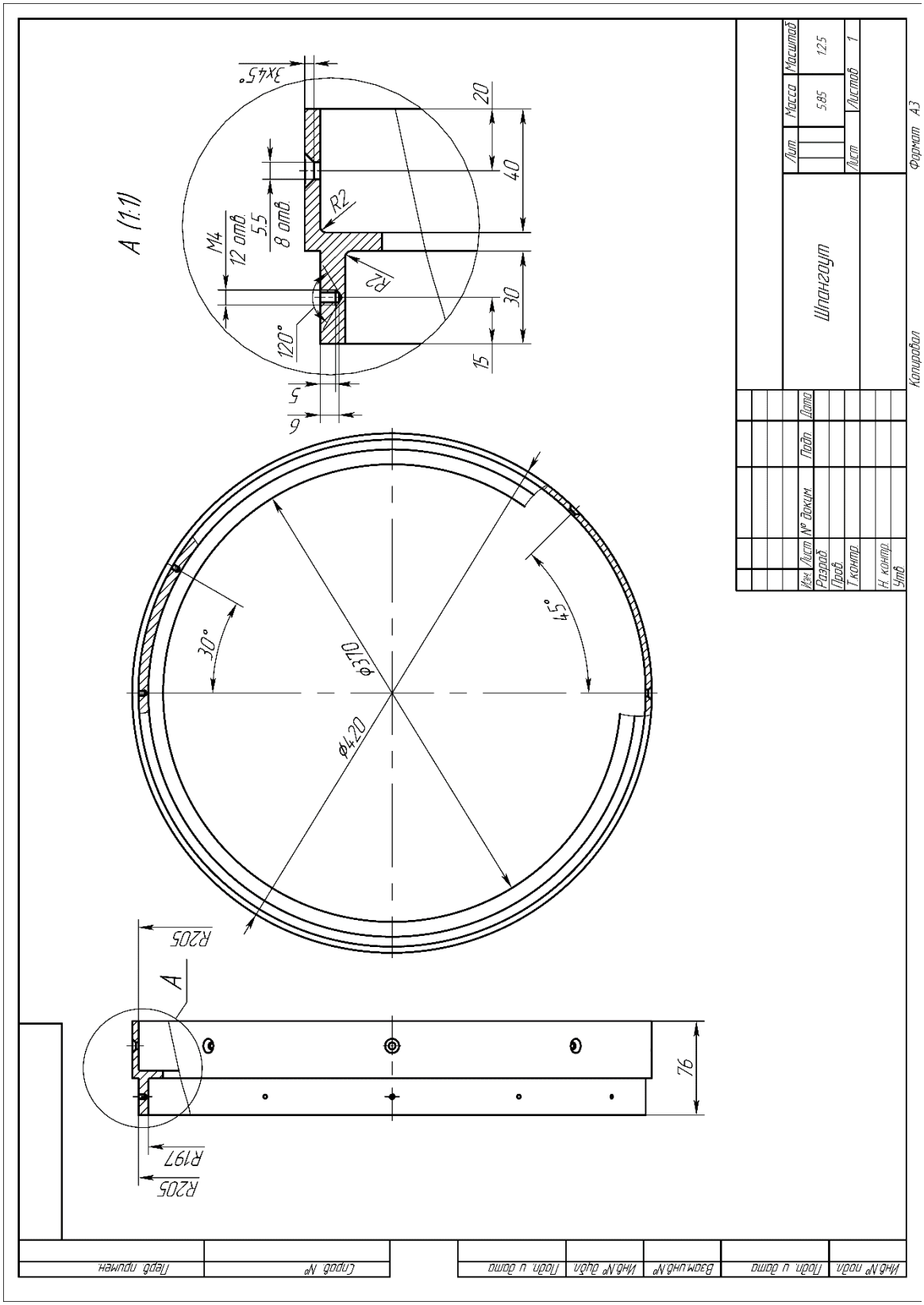
Ответ:



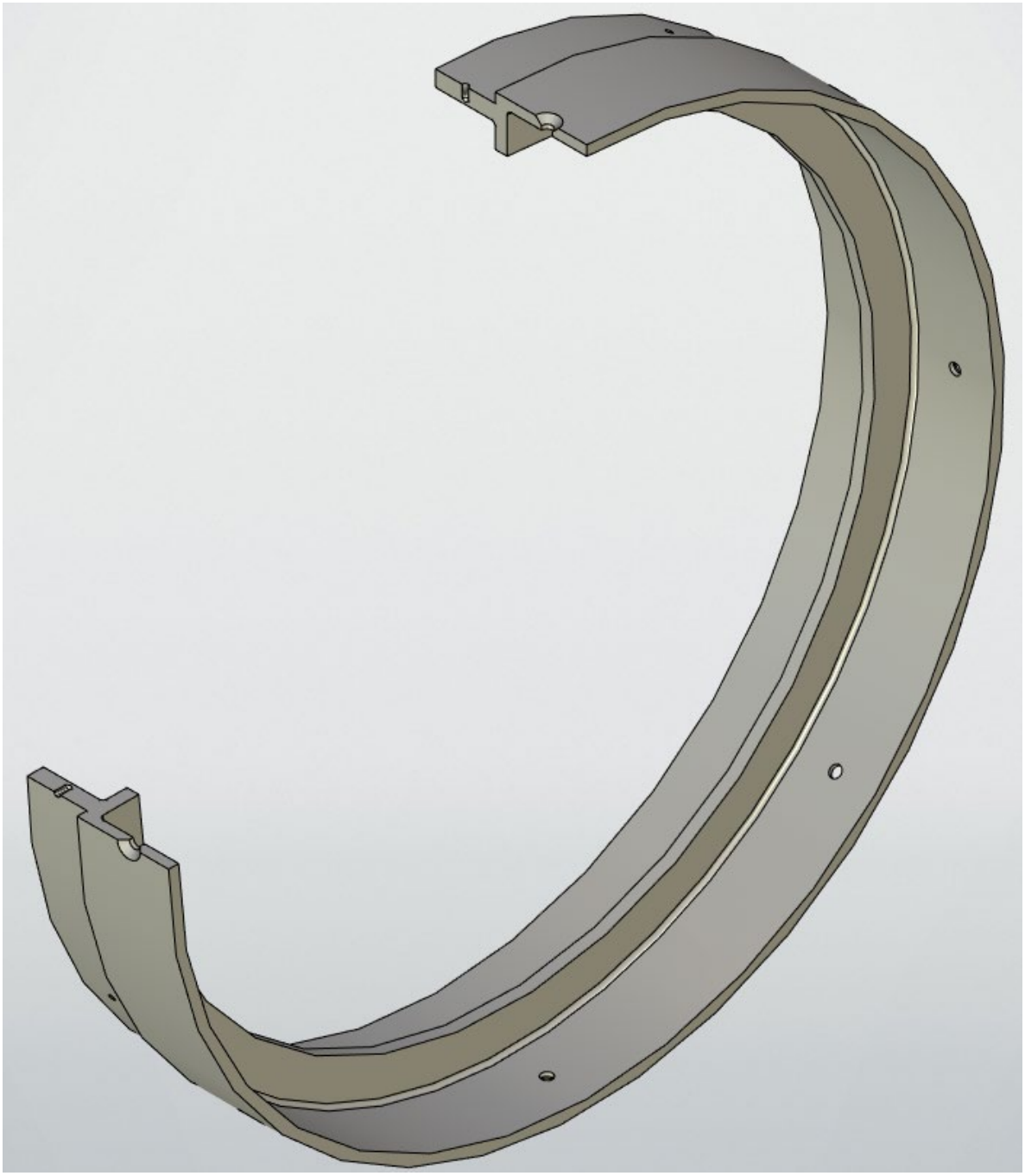
Рассечённая модель шпангоута в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



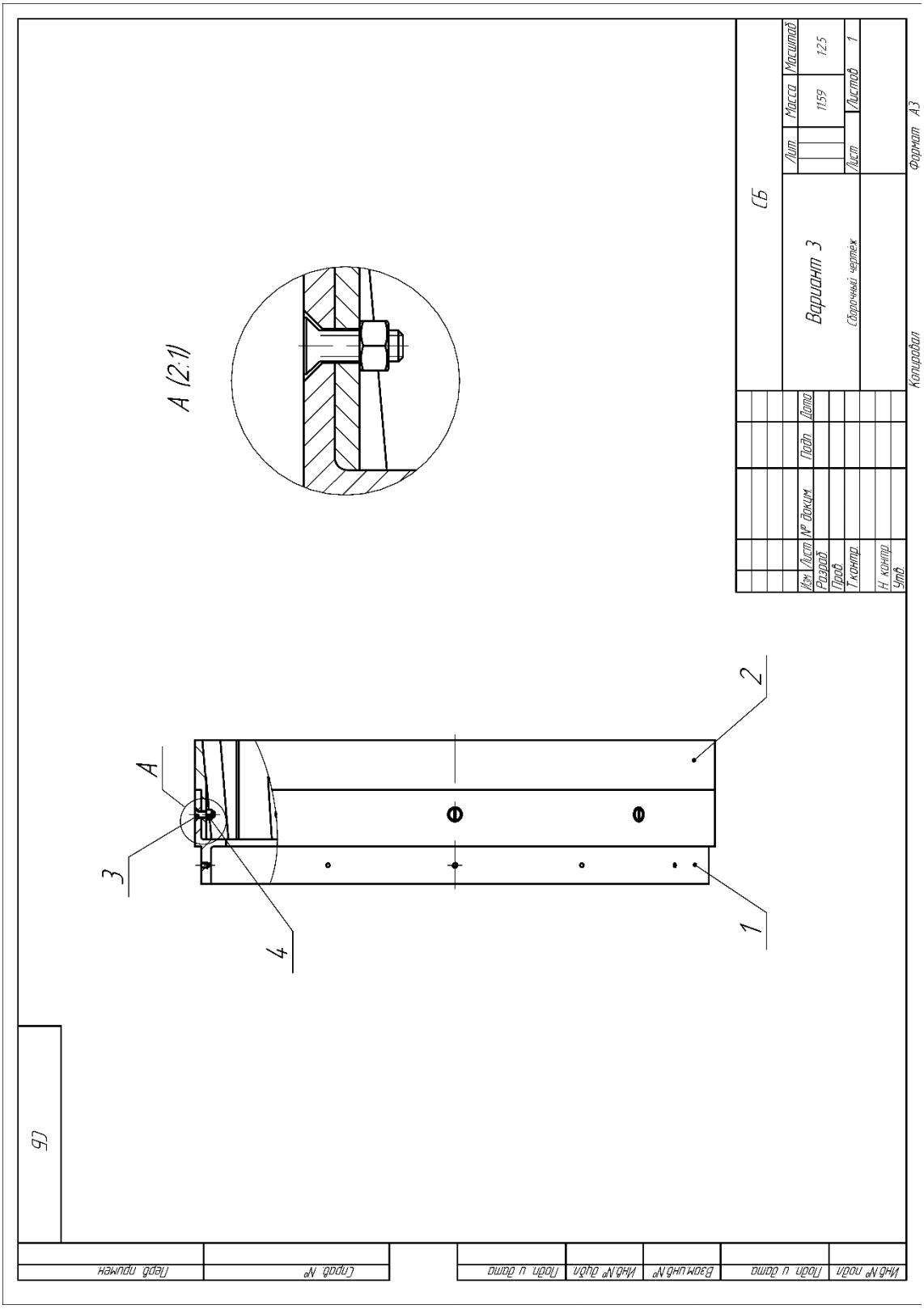
Ответ:



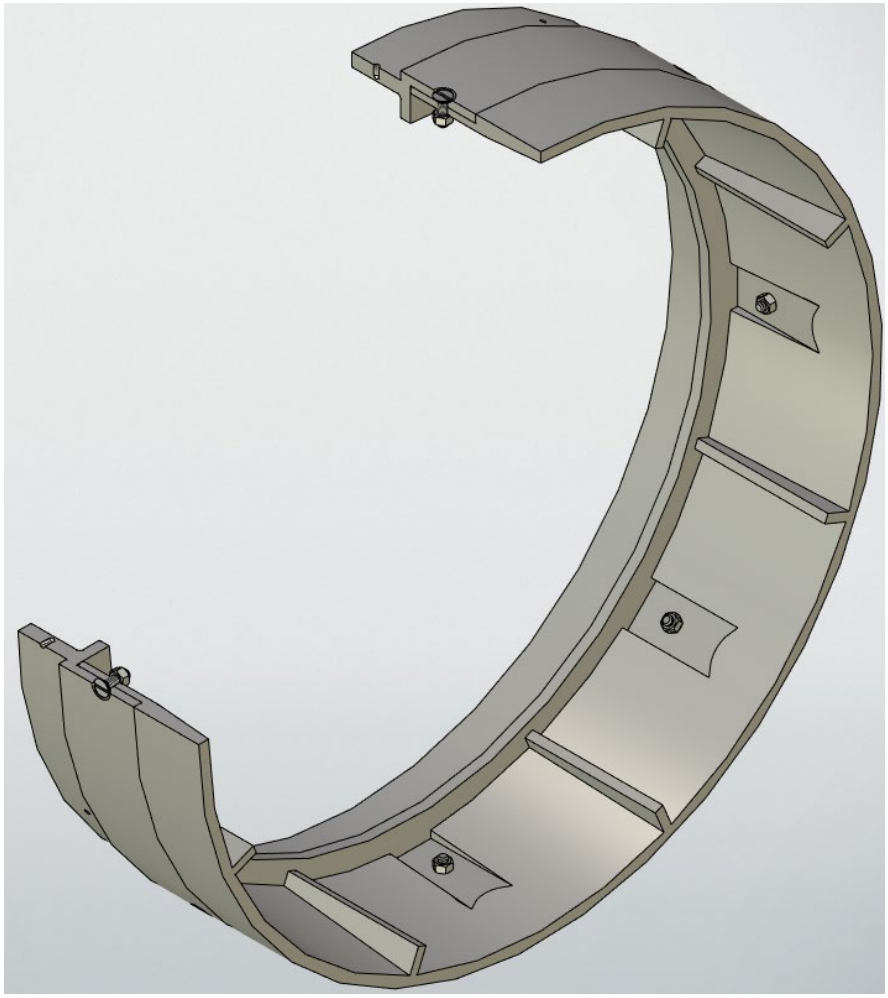
Рассечённая модель шпангоута в изометрии

Задание №3

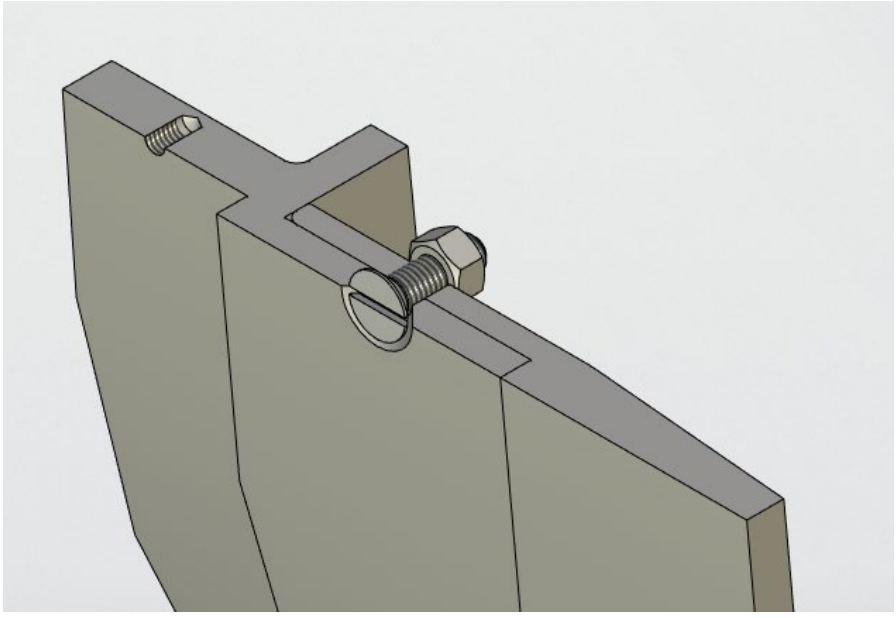
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Шпангоут	1	
А3		2		Шпангоут усиленный	1	
				Стандартные изделия		
		3		Винт М5×16		
				ГОСТ 17475-80	8	
		4		Гайка М5		
				ГОСТ 5927-70	8	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 3

Лит.	Лист	Листов
		1

Копировал

Формат А4

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

Вариант 4

Кейс №1 «Программирование»

«Наиболее частое простое число в матрице»

Разработать на языке Python программу, которая определяет, какое простое число встречается чаще всего в матрице.

Пользователь вводит размеры матрицы n и m, затем n × m целых положительных чисел. Требуется:

Сформировать матрицу n × m из введённых чисел;

Найти все простые числа в матрице и подсчитать частоту каждого;

Определить наиболее часто встречающееся простое число. Если таких несколько — выбрать наименьшее;

Вывести:

- Все найденные простые числа в матрице, отсортированные по возрастанию (метод пузырьковой сортировки);
- Наиболее частое простое число и его количество;

Если в матрице нет простых чисел — вывести сообщение: «Простых чисел не найдено.»

Таблица с функциями:

Функция	Параметры	Описание	Пример вызова функции
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
bubble_sort	массив (список): **list[int]**	Сортировка пузырьком по возрастанию	bubble_sort([5, 3, 8])
count_primes	matrix: list[list[int]]	Возвращает кортеж (primes, freq), где primes - список всех простых чисел, freq - словарь частот	primes, freq = count_primes(matrix)
print_frequent_prime	primes: list[int],	Вывод отсортированных простых чисел и наиболее частого	print_frequent_prime(primes: list[int], freq: dict[int, int])

	freq: dict[int, int]		

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размеры n и m от 1 до 100.
При равной частоте выбирается наименьшее по значению простое число.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются)
Функция is_prime(n) проверяет простоту через деление до √n.
Простые числа определяются вручную, без sympy и других библиотек.
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
При отсутствии подходящих чисел выводится сообщение: «Простых чисел не найдено»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:
def main():

```
n, m = map(int, input("Введите размеры матрицы: ").split())

print("Введите элементы матрицы построчно:")
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные
Пример 1: Простые в диапазоне от 5 до 13

Ввод:

3 3

2 3 5

3 6 3

7 3 11

Вывод:

Простые числа в матрице (отсортированы): [2, 3, 3, 3, 3, 5, 7, 11]

Наиболее частое простое число: 3 (встречается 4 раза)

Пример 2: Все простые по одному разу

Ввод:

2 3

2 3 5

7 11 13

Вывод:

Простые числа в матрице (отсортированы): [2, 3, 5, 7, 11, 13]

Наиболее частое простое число: 2 (встречается 1 раз)

Пример 3: Простые повторяются, но две с одинаковой частотой

Ввод:

2 4

2 3 2 3

5 5 7 7

Вывод:

Простые числа в матрице (отсортированы): [2, 2, 3, 3, 5, 5, 7, 7]

Наиболее частое простое число: 2 (встречается 2 раза)

Пример 4: Только одно простое число

Ввод:

2 2

4 6

8 7

Вывод:

Простые числа в матрице (отсортированы): [7]

Наиболее частое простое число: 7 (встречается 1 раз)

Пример 5: Нет простых чисел

Ввод:

2 2

4 6

8 9

Вывод:

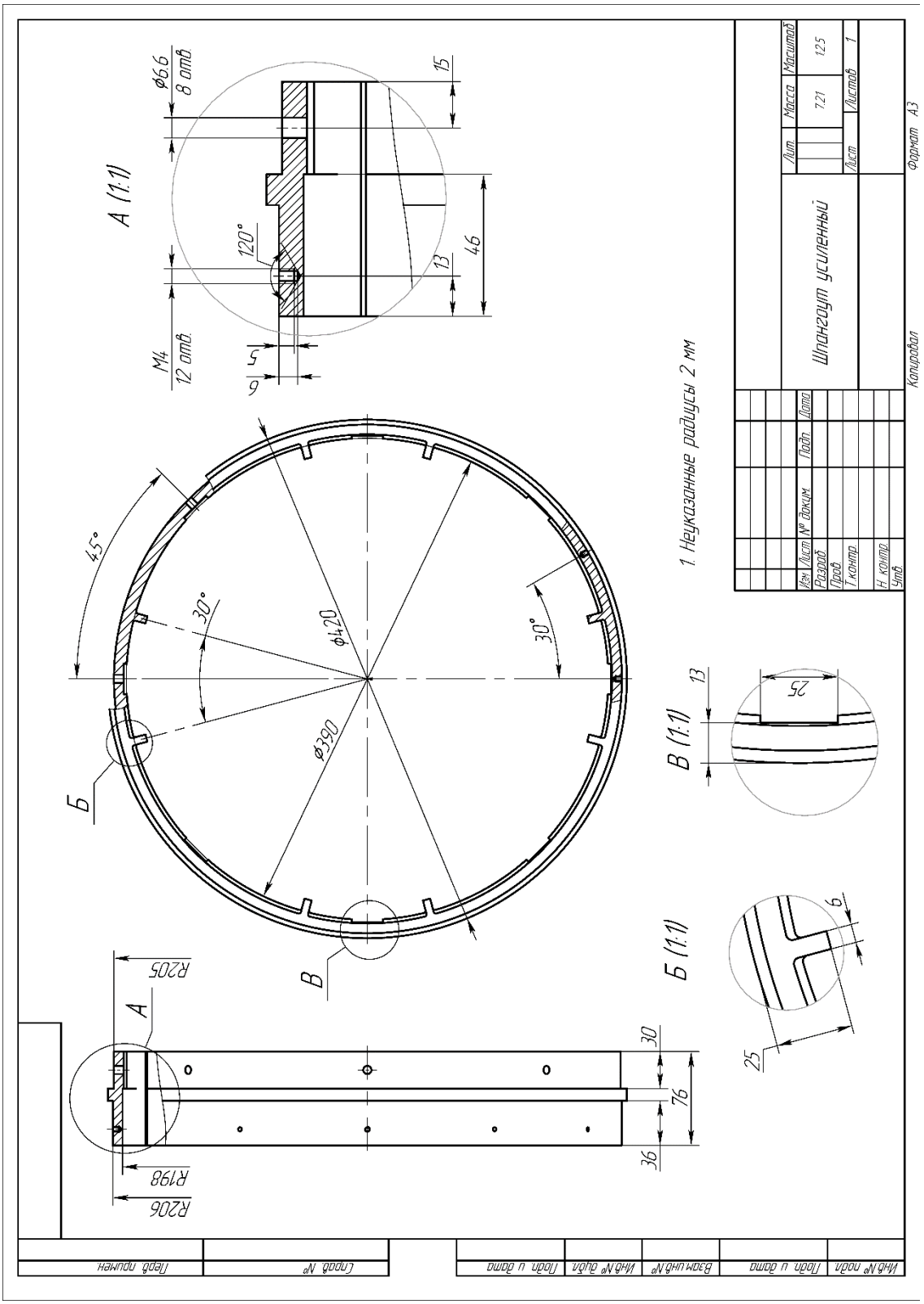
Простых чисел не найдено.

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

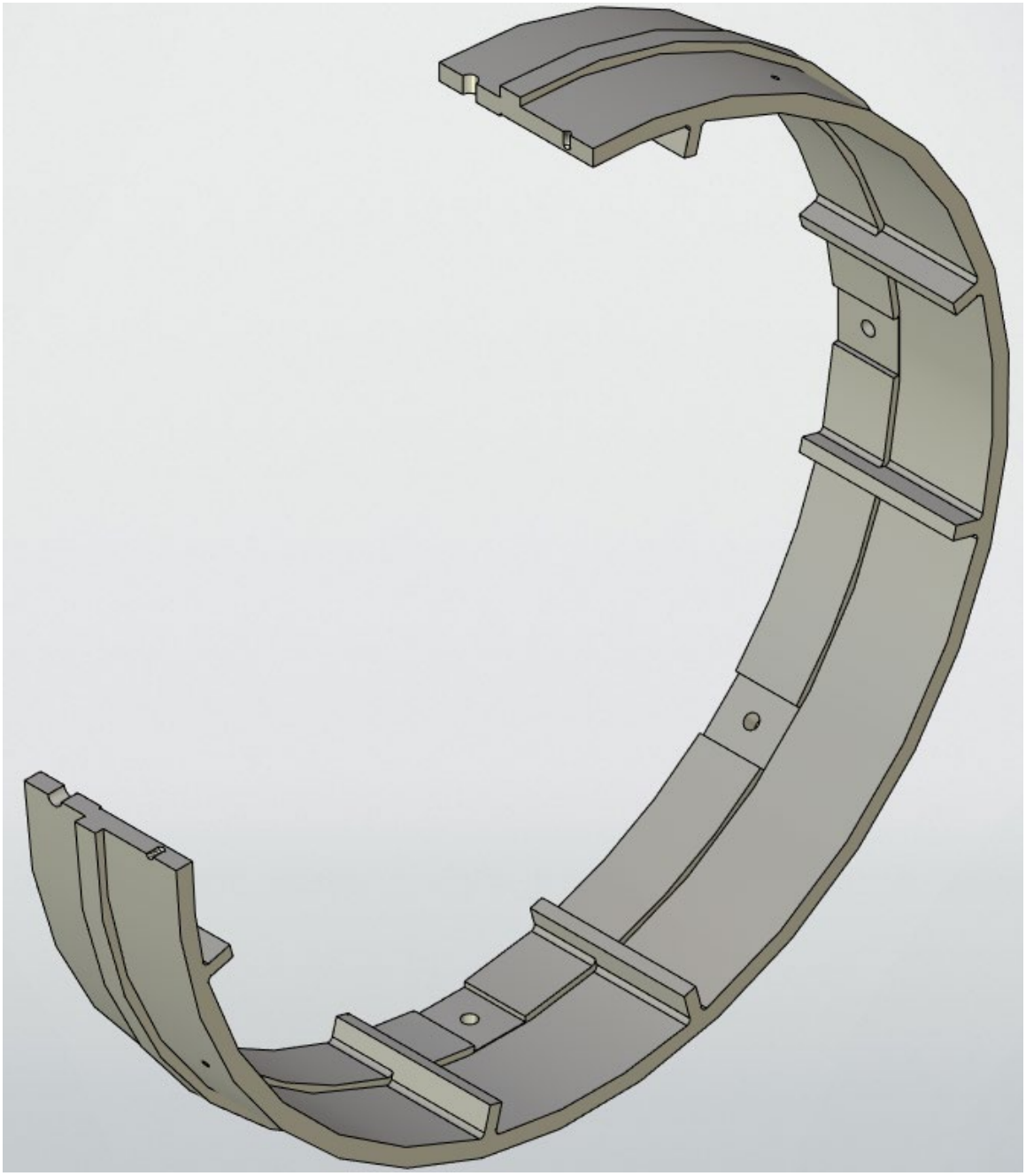
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



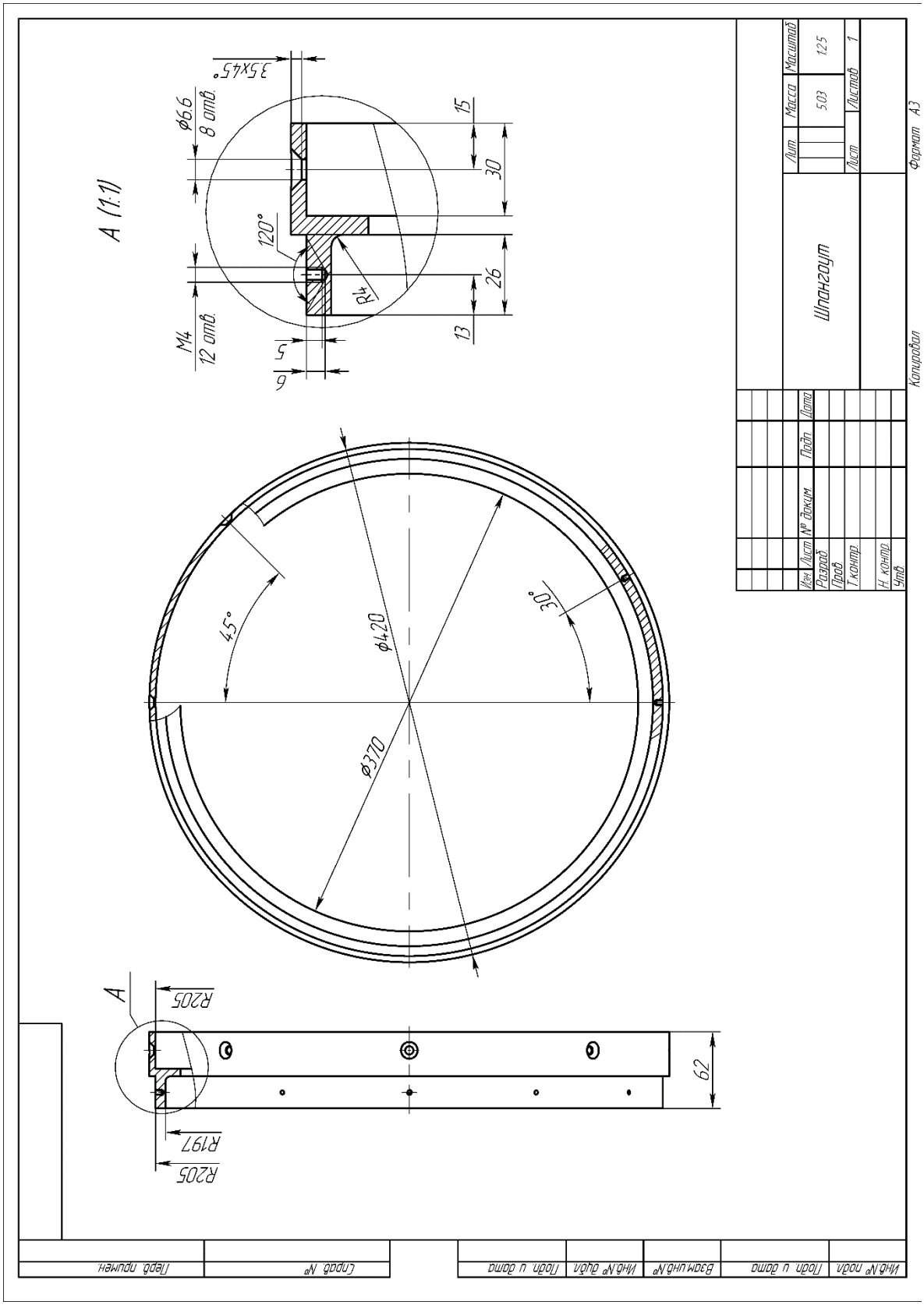
Ответ:



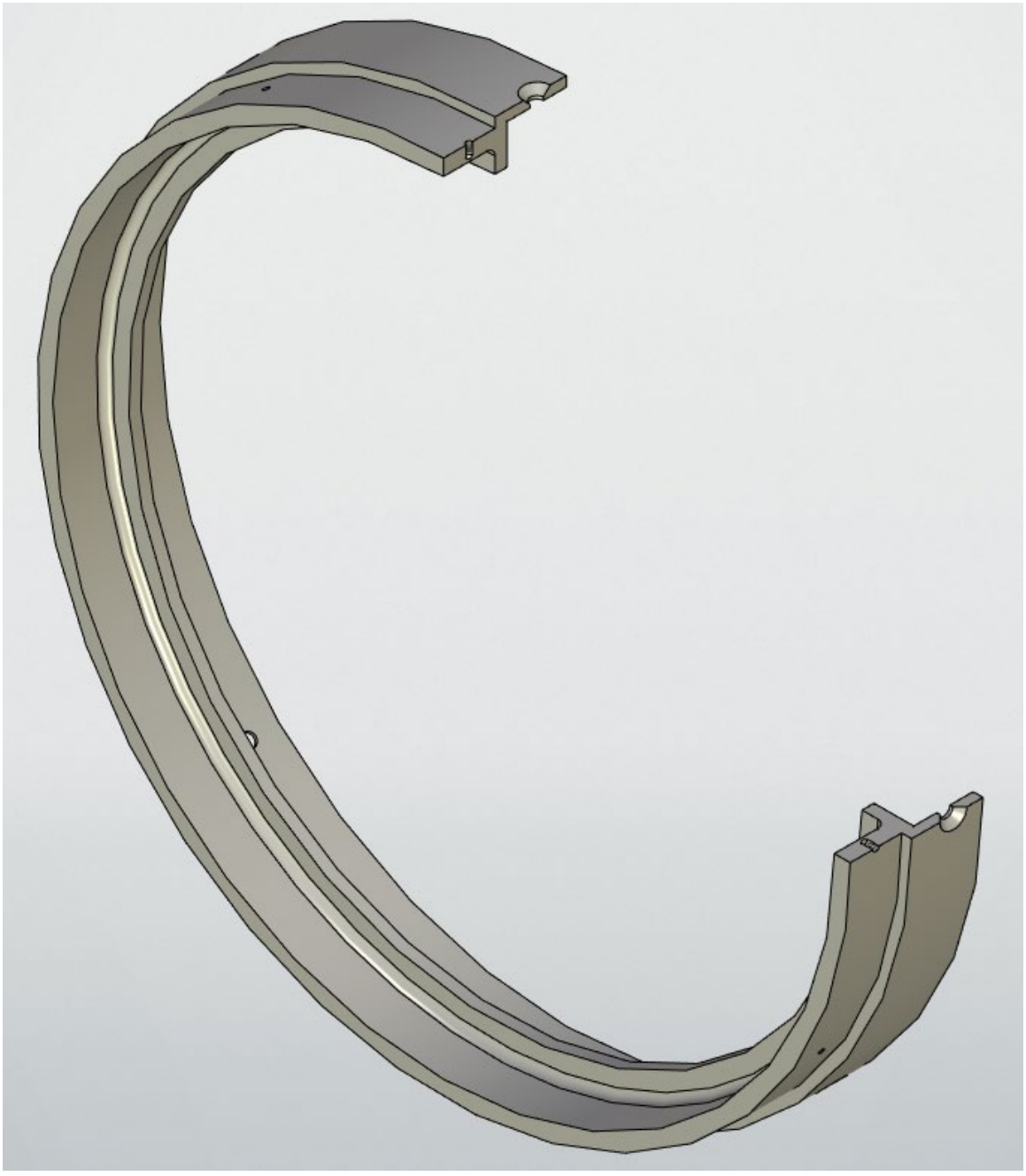
Рассечённая модель шпангоута в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



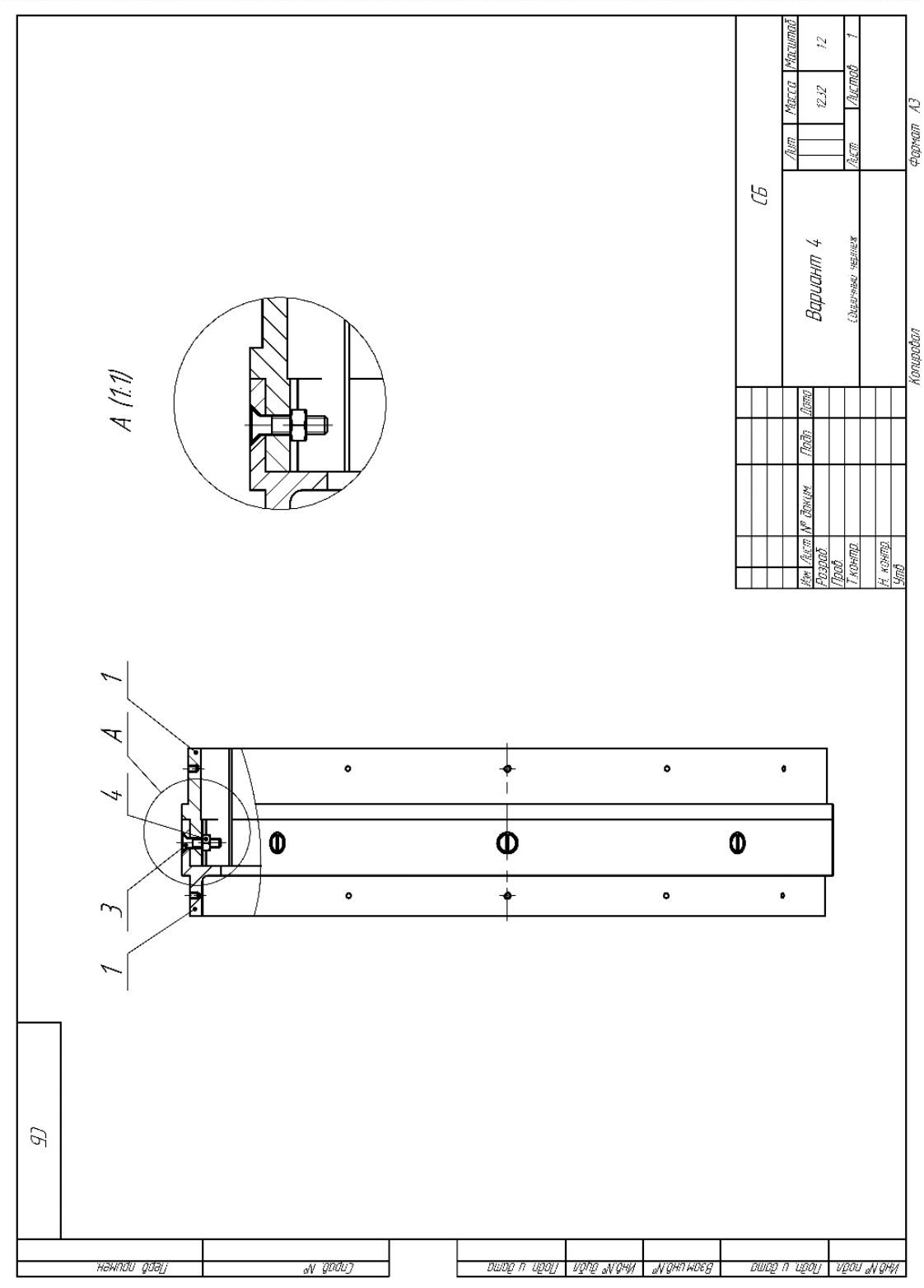
Ответ:



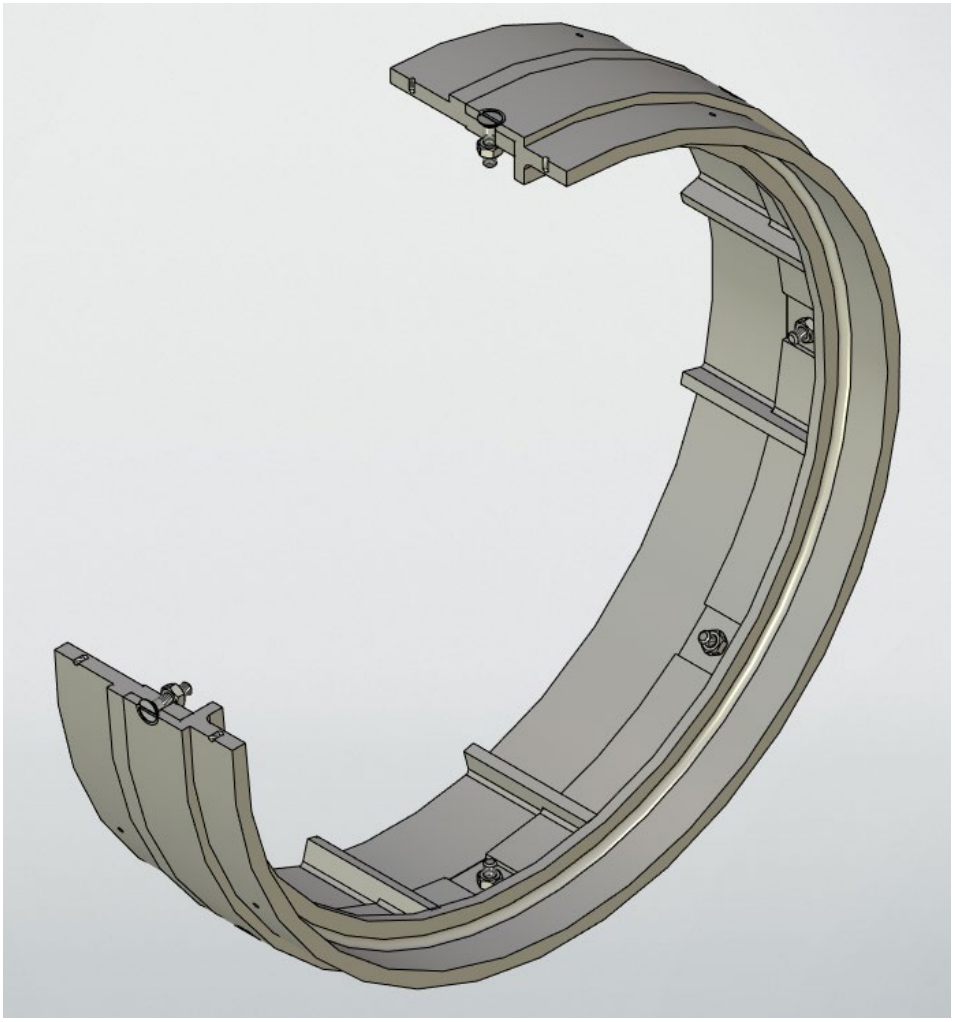
Рассечённая модель шпангоута в изометрии

Задание №3

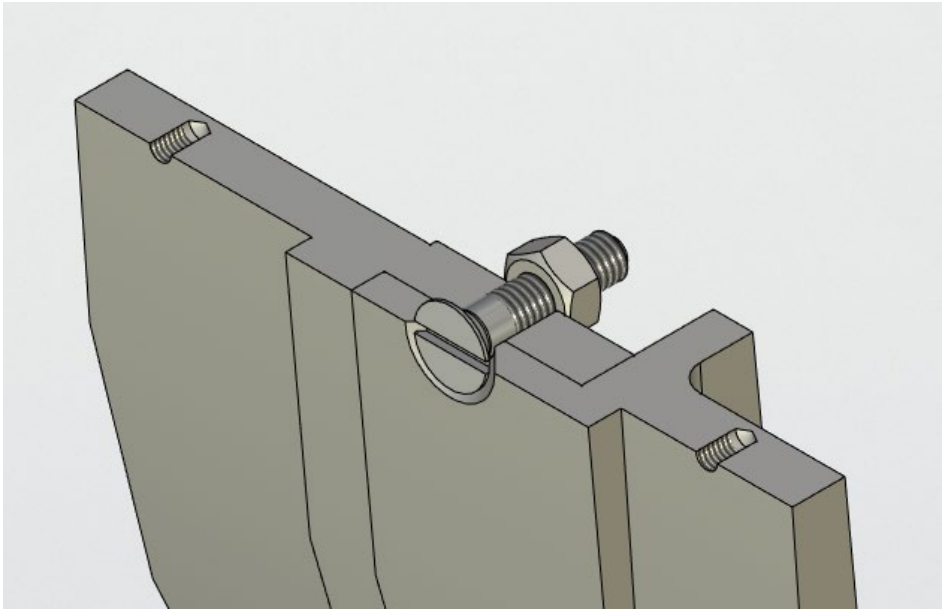
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Шпангоут	1	
А3		2		Шпангоут усиленный	1	
				Стандартные изделия		
		3		Винт М6×25		
				ГОСТ 17475-80	8	
		4		Гайка М6		
				ГОСТ 5927-70	8	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 4

Лит.	Лист	Листов
		1

Копировал

Формат А4

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

Вариант 5

Кейс №1 «Программирование»

«Максимальная плотность простых в подматрице»

Разработать на языке Python программу, которая находит подматрицу фиксированного размера k × k с наибольшей плотностью простых чисел.

Пользователь вводит размеры матрицы n и m, затем n × m целых положительных чисел. Затем вводится размер подматрицы k (целое число ≥1 и ≤ min(n, m)). Гарантируется, что n, m ≤ 50 при k ≥ 10. Требуется:

Сформировать матрицу n × m из введённых чисел;

Для всех возможных подматриц размером k × k определить долю простых чисел в ней (число простых / общее количество k × k);

Найти подматрицу с максимальной долей простых чисел;

Вывести:

- Координаты подматрицы: (верхняя строка, левый столбец) до (нижняя строка, правый столбец);
- Список всех простых чисел внутри найденного подматрицы (отсортированных по возрастанию с помощью пузырьковой сортировки);
- Количество простых чисел и процент плотности (в процентах, округлённый до двух знаков);

Если ни в одной подматрице нет простых чисел — вывести: «Простых чисел в подматрицах не найдено.»

Таблица с функциями:

Функция	**Параметры**	**Описание**	**Пример вызова функции**
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
bubble_sort	массив (список): **list[int]**	Сортировка пузырьком по возрастанию	bubble_sort([5, 3, 8])

find_max_density_window	matrix: list[list[int]], k: int	Поиск окна с максимальной плотностью простых чисел	find_max_density_window(matrix, 2)
print_density_info	primes: list[int], coords: tuple[int, int, int, int], percent: float	Вывод координат, списка и плотности	print_density_info(primes, (0,0,1,1), 75.0)

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размеры n и m от 1 до 100.
Размер подматрицы k - целое число, $1 \leq k \leq \min(n, m)$.
Для случаев, когда $k \geq 10$, дополнительно гарантируется, что $n \leq 50$ и $m \leq 50$.
При равной частоте выбирается наименьшее по значению простое число.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются).
При равной плотности — берётся подматрица с минимальными верхними координатами.
Функция is_prime(n) проверяет простоту через деление до $\sqrt{n}$.
Простые числа определяются вручную, без sympy и других библиотек.
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
При отсутствии подходящих чисел выводится сообщение: «Простых чисел в подматрицах не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:

```
def main():

    n, m = map(int, input("Введите размеры матрицы: ").split())

    print("Введите элементы матрицы построчно:")

    ... #Ввод элементов построчно
```

```
    k = int(input("Введите размер окна k: "))
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные

Пример 1: Максимальная плотность 75%

Ввод:

3 3

2 3 4

5 6 7

8 9 10

2

Вывод:

Подматрица с максимальной плотностью простых чисел: (0, 0) до (1, 1)

Простые числа подматрицы (отсортированы): [2, 3, 5]

Количество простых: 3 из 4 (75.00%)

Пример 2: Все подматрицы с нулевой плотностью

Ввод:

2 2

4 6

8 10

1

Вывод:

Простых чисел в подматрицах не найдено.

Пример 3: Полностью простая подматрица

Ввод:

2 2

2 3

5 7

2

Вывод:

Подматрица с максимальной плотностью простых чисел: (0, 0) до (1, 1)

Простые числа подматрицы (отсортированы): [2, 3, 5, 7]

Количество простых: 4 из 4 (100.00%)

Пример 4: Несколько подматриц с одинаковой плотностью — выбирается первая

Ввод:

3 3

2 4 2

6 2 6

2 6 2

2

Вывод:

Подматрица с максимальной плотностью простых чисел: (0, 0) до (1, 1)

Простые числа подматрицы (отсортированы): [2, 2]

Количество простых: 2 из 4 (50.00%)

Пример 5: Подматрицы 1×1, выбирается с наименьшим числом

Ввод:

2 2

3 5

7 11

1

Вывод:

Подматрица с максимальной плотностью простых чисел: (0, 0) до (0, 0)

Простые числа подматрицы (отсортированы): [3]

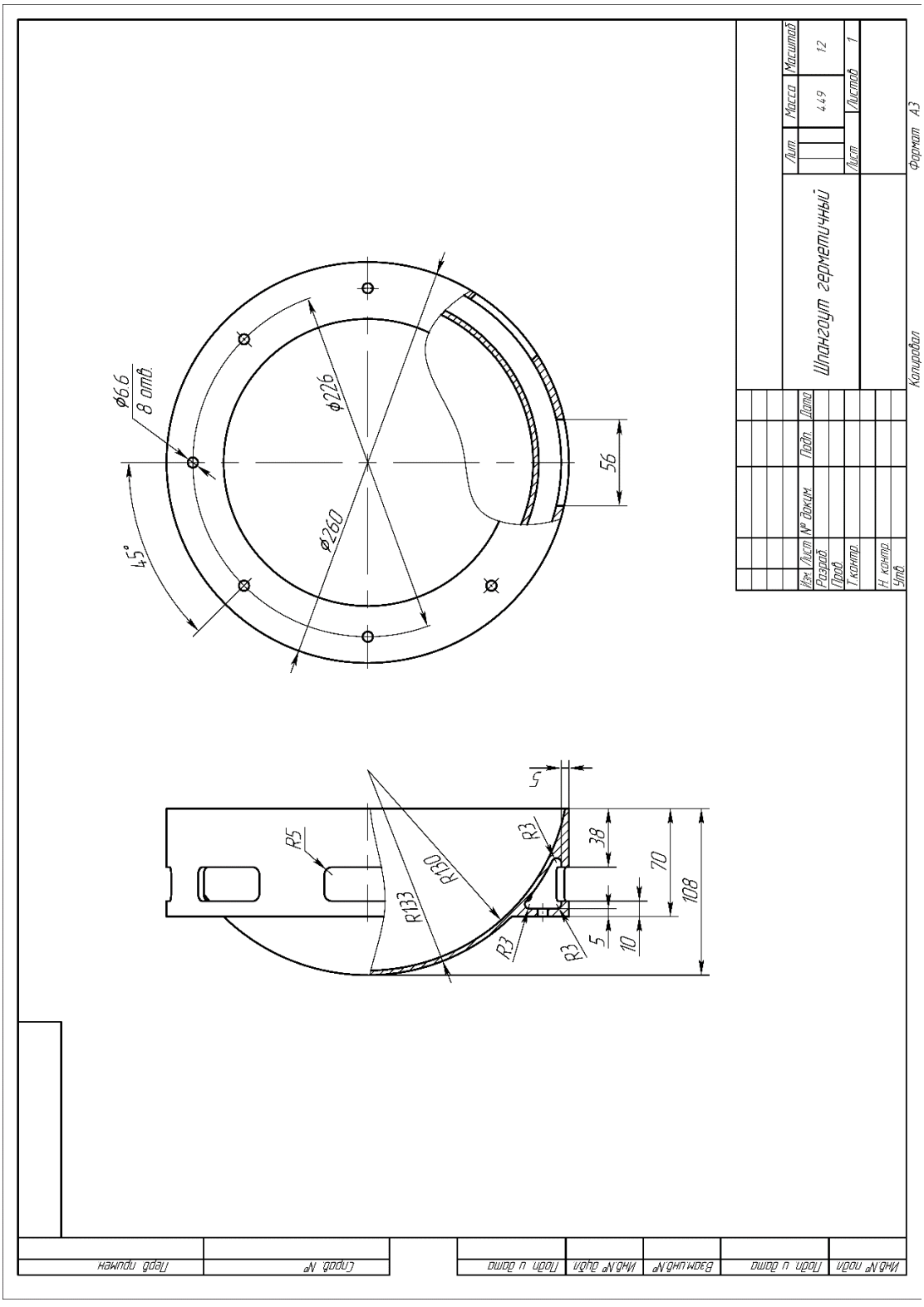
Количество простых: 1 из 1 (100.00%)

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

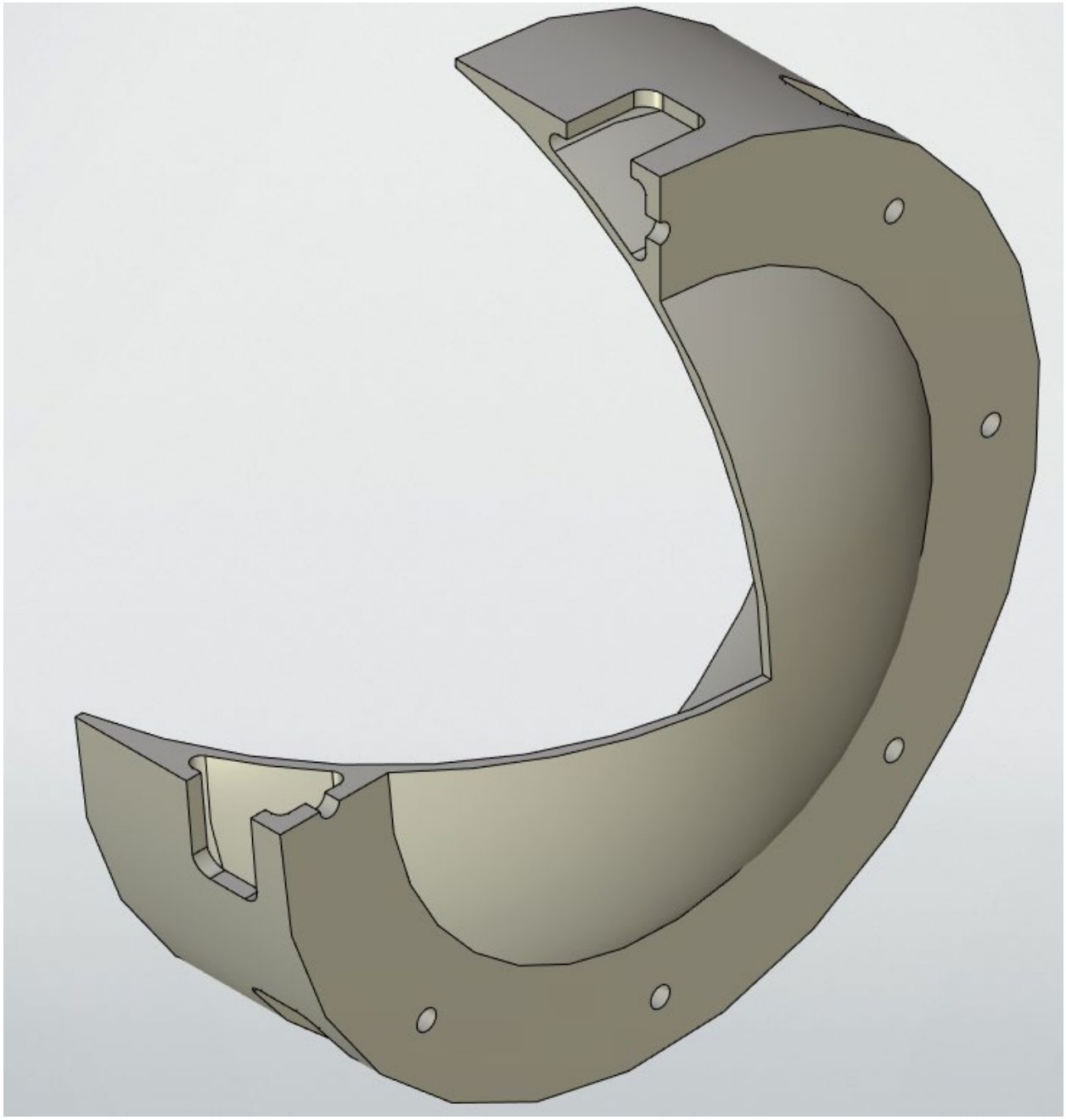
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



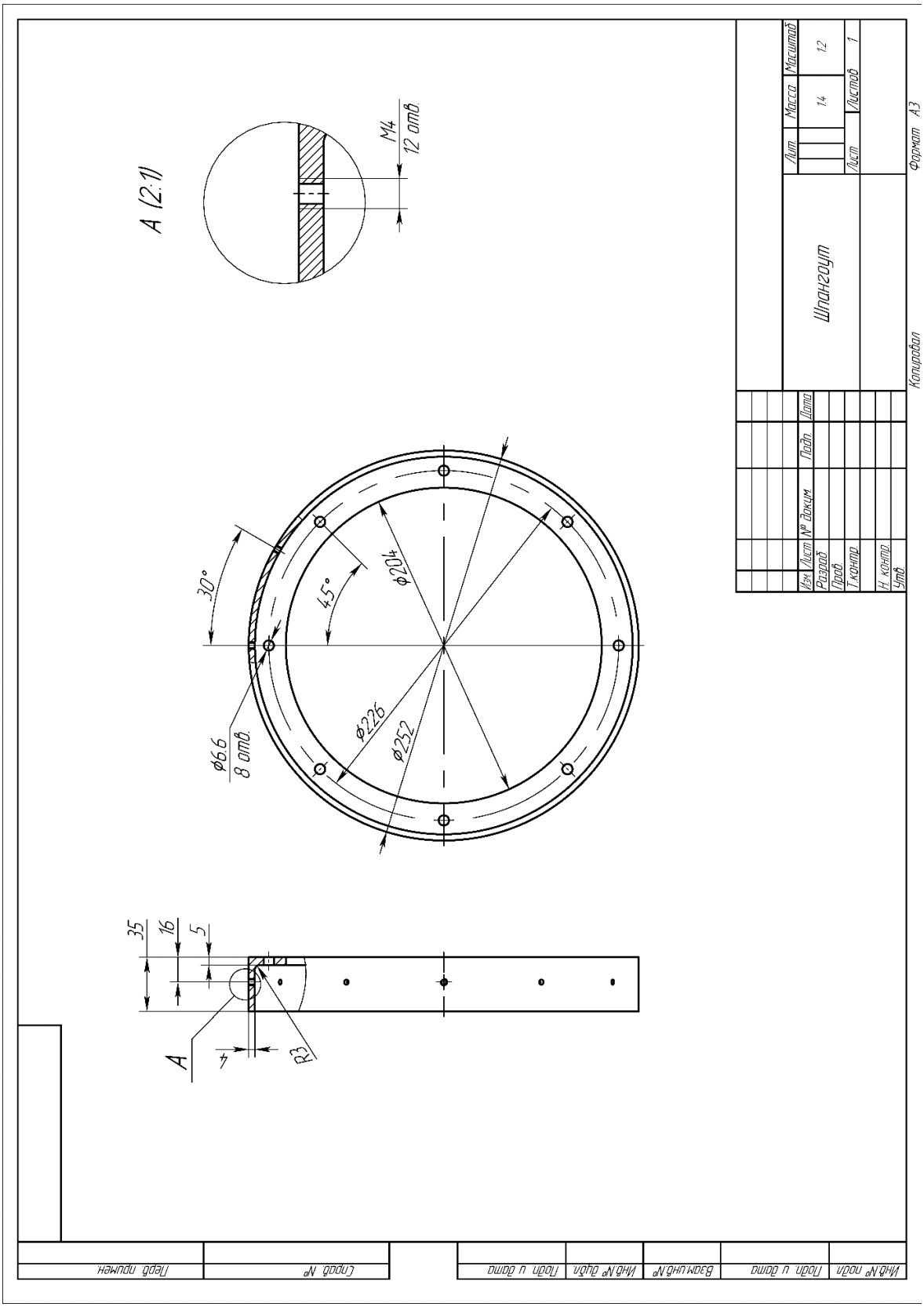
Ответ:



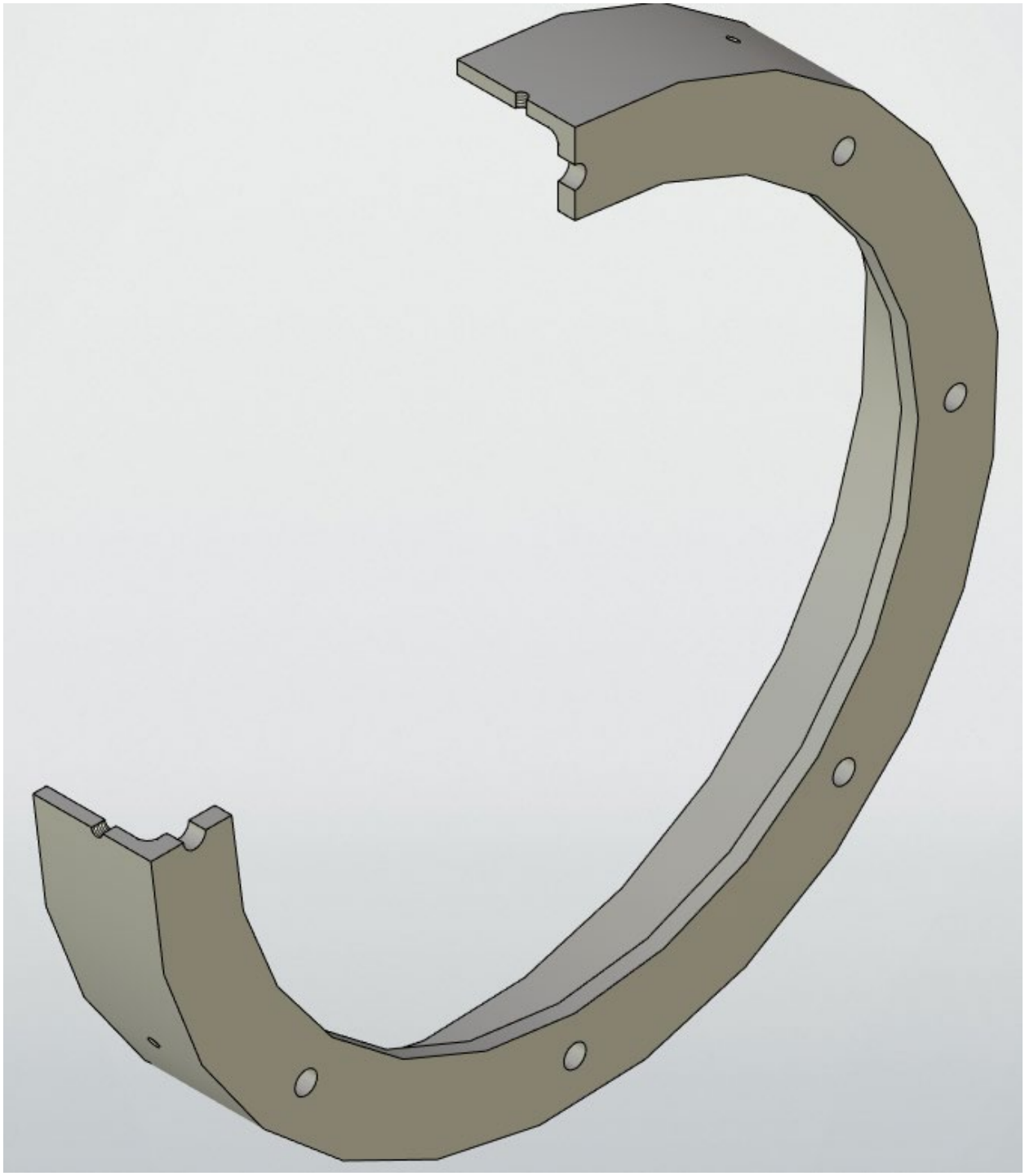
Рассечённая модель шпангоута в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



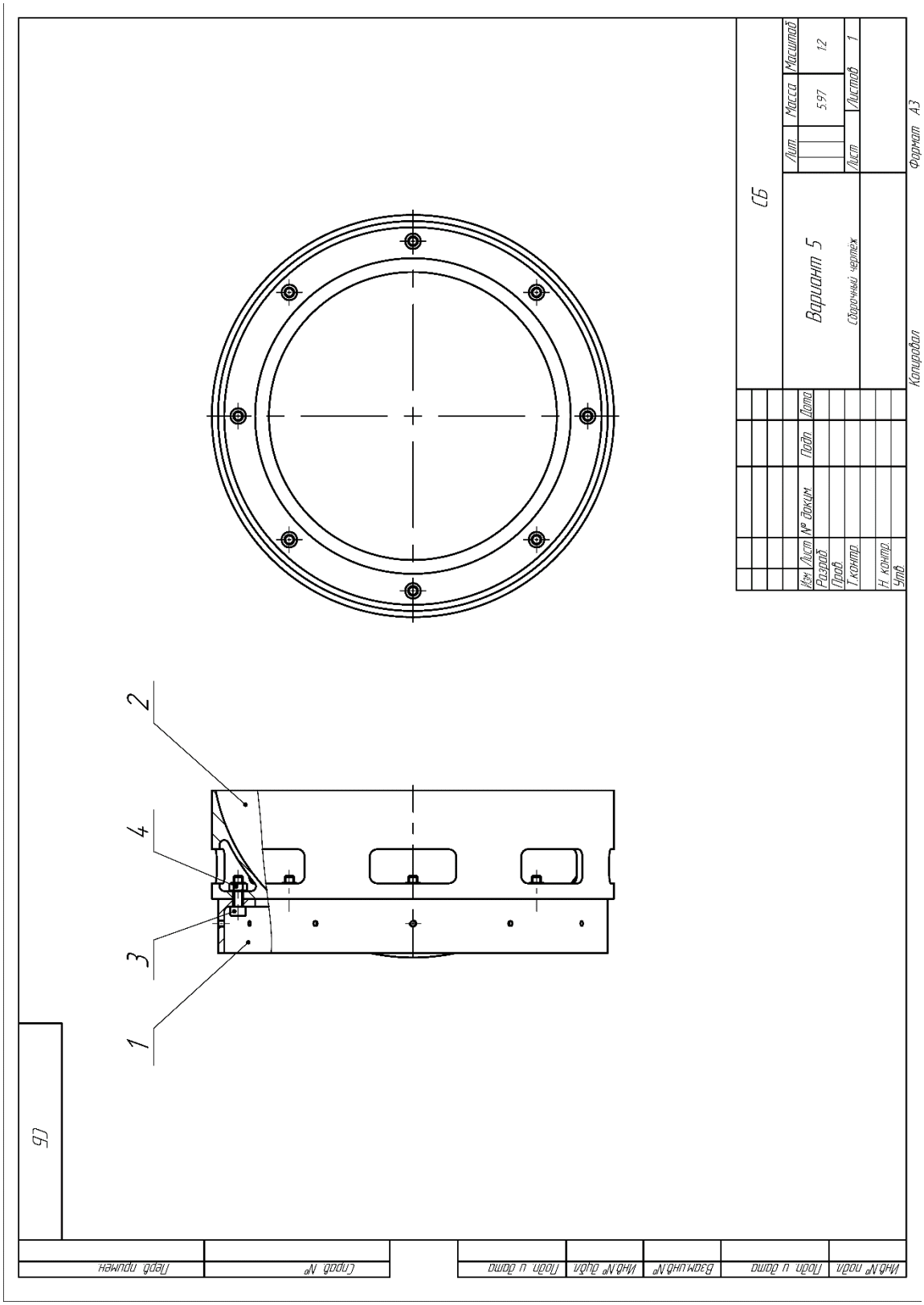
Ответ:



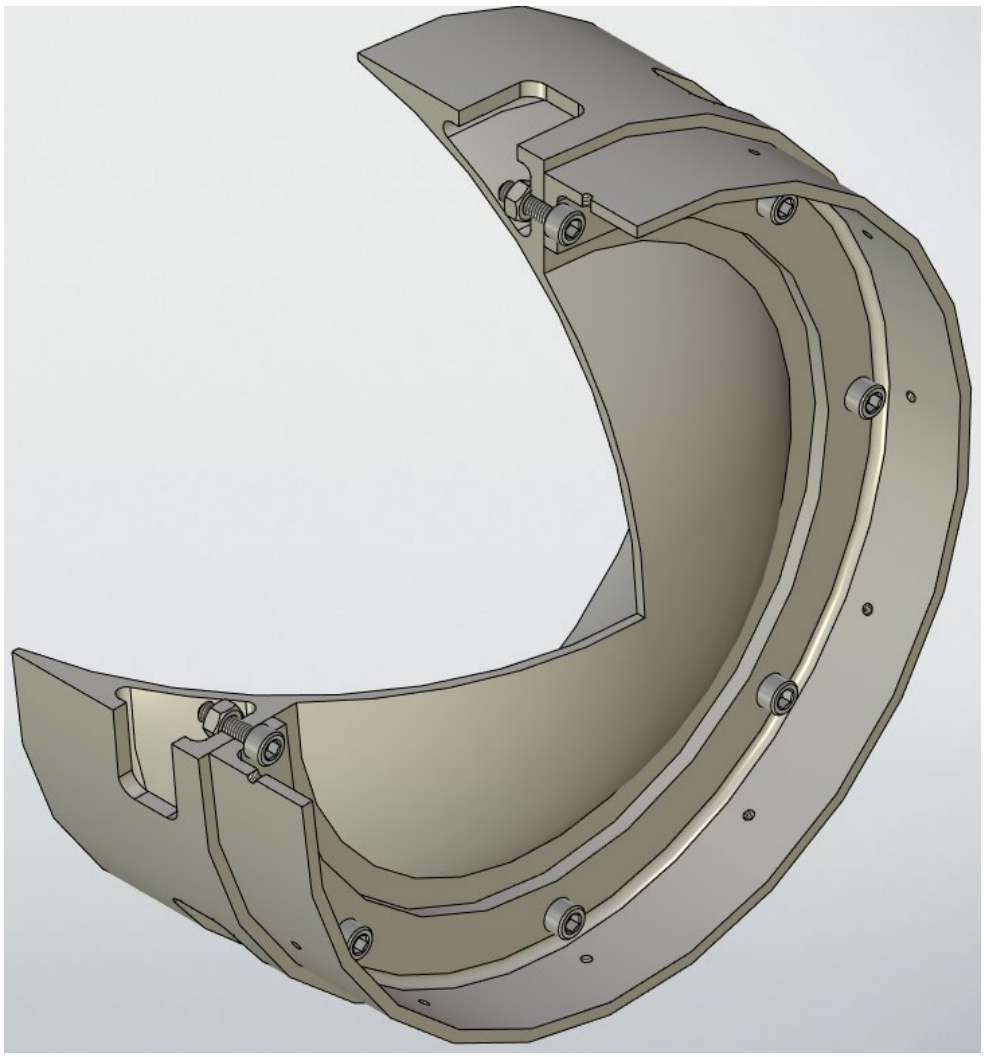
Рассечённая модель шпангоута в изометрии

Задание №3

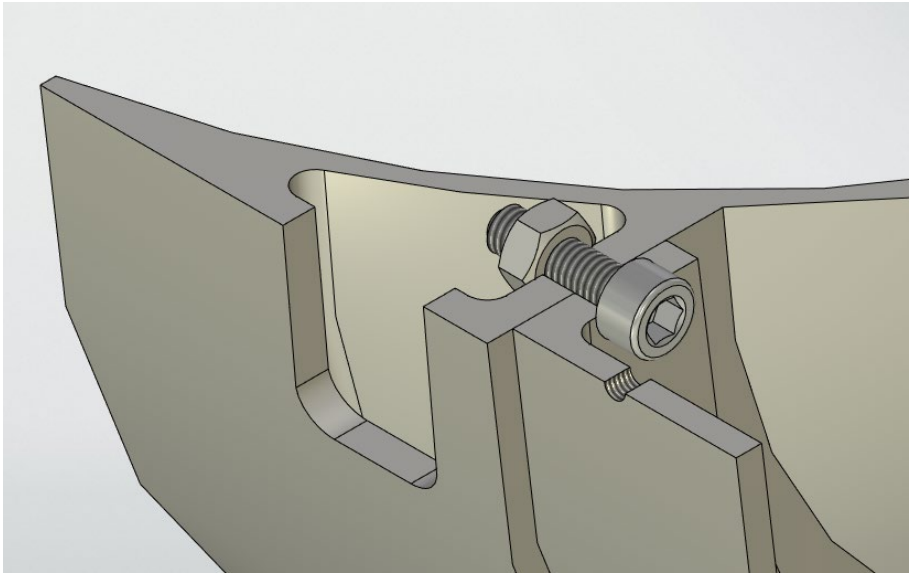
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Шпангоут	1	
А3		2		Шпангоут герметичный	1	
				Стандартные изделия		
		3		Винт М6×20 ГОСТ 11738-84	8	
		4		Гайка М6 ГОСТ 5927-70	8	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 5

Лит.	Лист	Листов
		1

Копировал

Формат А4

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

Вариант 6

Кейс №1 «Программирование»

«Симметрия простых по главной диагонали»

Разработать на языке Python программу, которая проверяет, является ли матрица симметричной по главной диагонали среди простых чисел.

Пользователь вводит размер квадратной матрицы n, затем n × n целых положительных чисел. Требуется:

Сформировать квадратную матрицу n × n из введённых чисел;

Определить, какие элементы в матрице являются простыми;

Проверить, является ли матрица симметричной по главной диагонали, только в тех позициях, где стоят простые числа:

- Элемент A[i][j] и A[j][i] должны оба быть простыми и равными;
- Остальные элементы (непростые) не проверяются;
- Элементы на диагонали (i=j) проверяются только на простоту (не требуют симметричного парного элемента)

Вывести:

- «Матрица симметрична по простым числам.» — если условие выполнено;
- «Матрица несимметрична по простым числам.» — если хотя бы одно несоответствие найдено;
- Если в матрице нет простых чисел — вывести: «Простых чисел не найдено.»

Таблица с функциями:

Функция	**Параметры**	**Описание**	**Пример вызова функции**
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
is_prime_symmetric	matrix: list[list[int]]	Проверка симметрии по диагонали для	is_prime_symmetric(matrix)

		простых чисел	
print_symmetry_result	result: str	Вывод результата симметрии	print_symmetry_result("симметричная")

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размер n от 1 до 100.
При равной частоте выбирается наименьшее по значению простое число.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются).
При равной плотности — берётся подматрица с минимальными верхними координатами.
Функция is_prime(n) проверяет простоту через деление до √n.
Простые числа определяются вручную, без sympy и других библиотек.
Если хотя бы один элемент A[i][j] ≠ A[j][i] при том, что оба простые — симметрия нарушена.
Если один из двух (A[i][j] или A[j][i]) не является простым — пара игнорируется.
При отсутствии подходящих чисел выводится сообщение: «Простых чисел не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:
def main():

```
n = int(input("Введите размер квадратной матрицы: "))

print("Введите элементы матрицы построчно:")
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные

Пример 1: Матрица симметрична по простым

Ввод:

3

2 4 3

4 5 6

3 6 7

Вывод:

Матрица симметрична по простым числам.

Пример 2: Матрица несимметрична по простым

Ввод:

3

2 3 4

7 5 6

2 6 11

Вывод:

Матрица несимметрична по простым числам.

Пример 3: Матрица состоит только из непростых

Ввод:

2

4 6

8 9

Вывод:

Простых чисел не найдено.

Пример 4: Один элемент — на диагонали

Ввод:

1

13

Вывод:

Матрица симметрична по простым числам.

Пример 5: Частичная симметрия, но не полностью

Ввод:

3

2 4 3

4 2 4

3 4 5

Вывод:

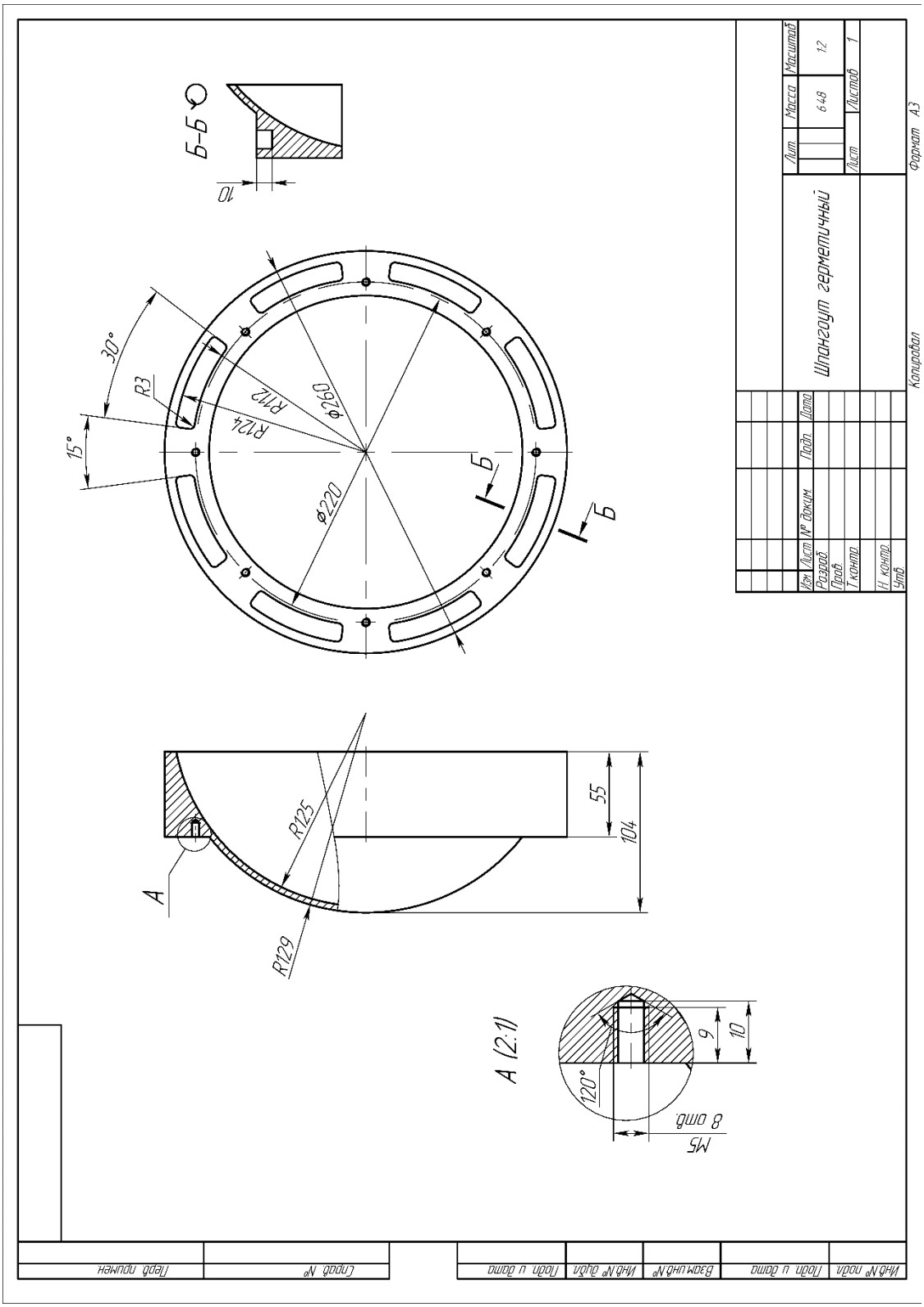
Матрица симметрична по простым числам.

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

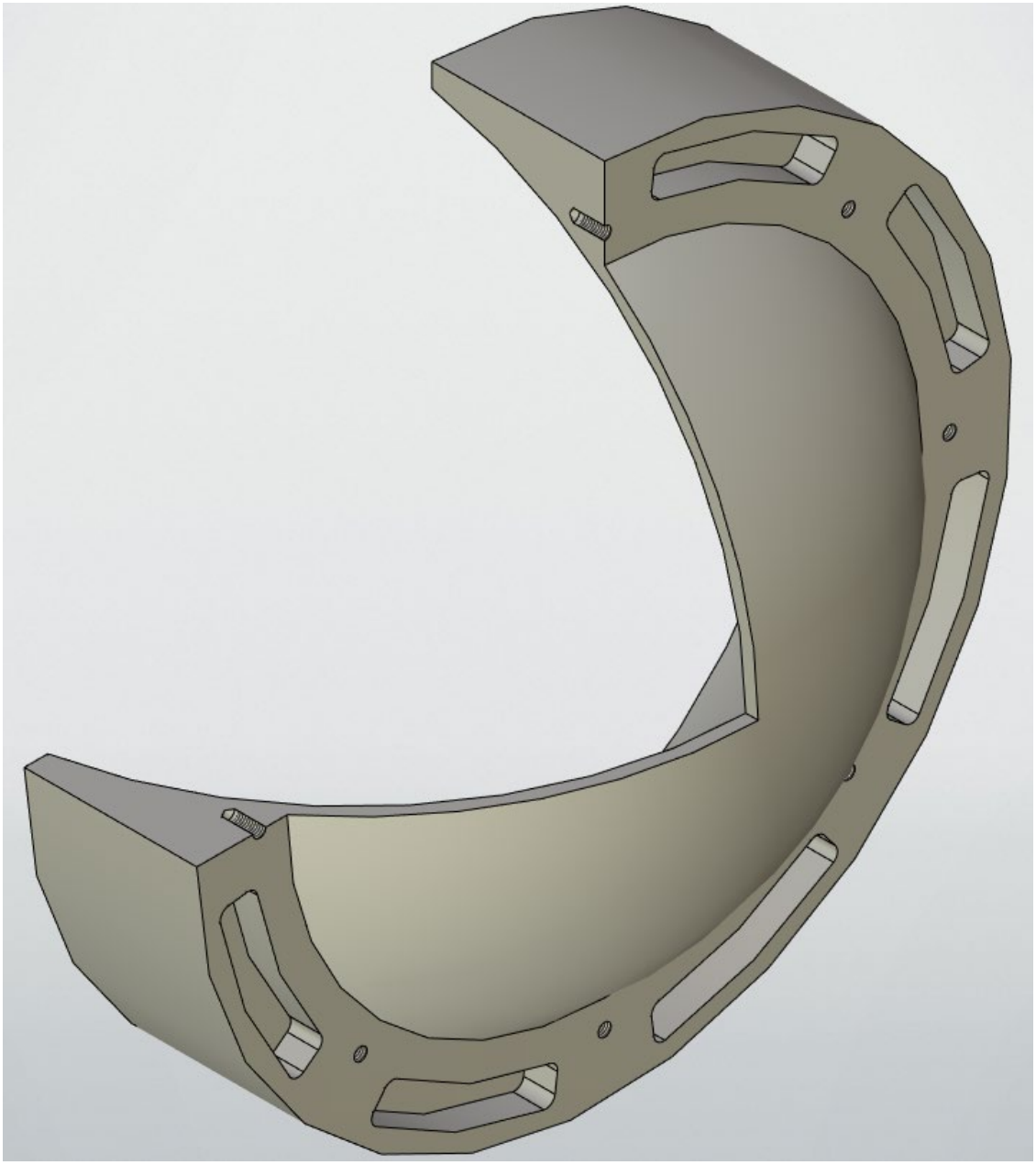
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



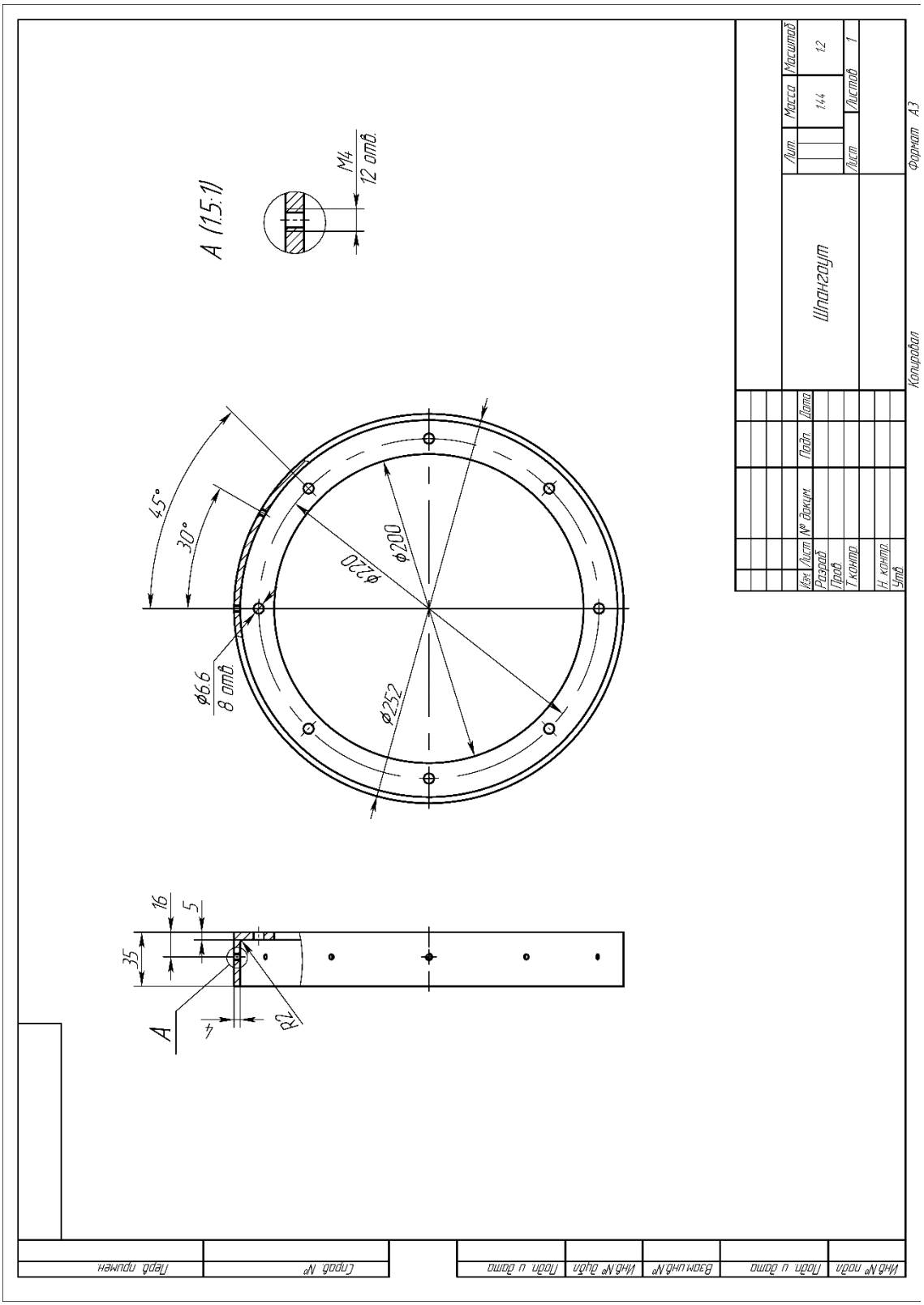
Ответ:



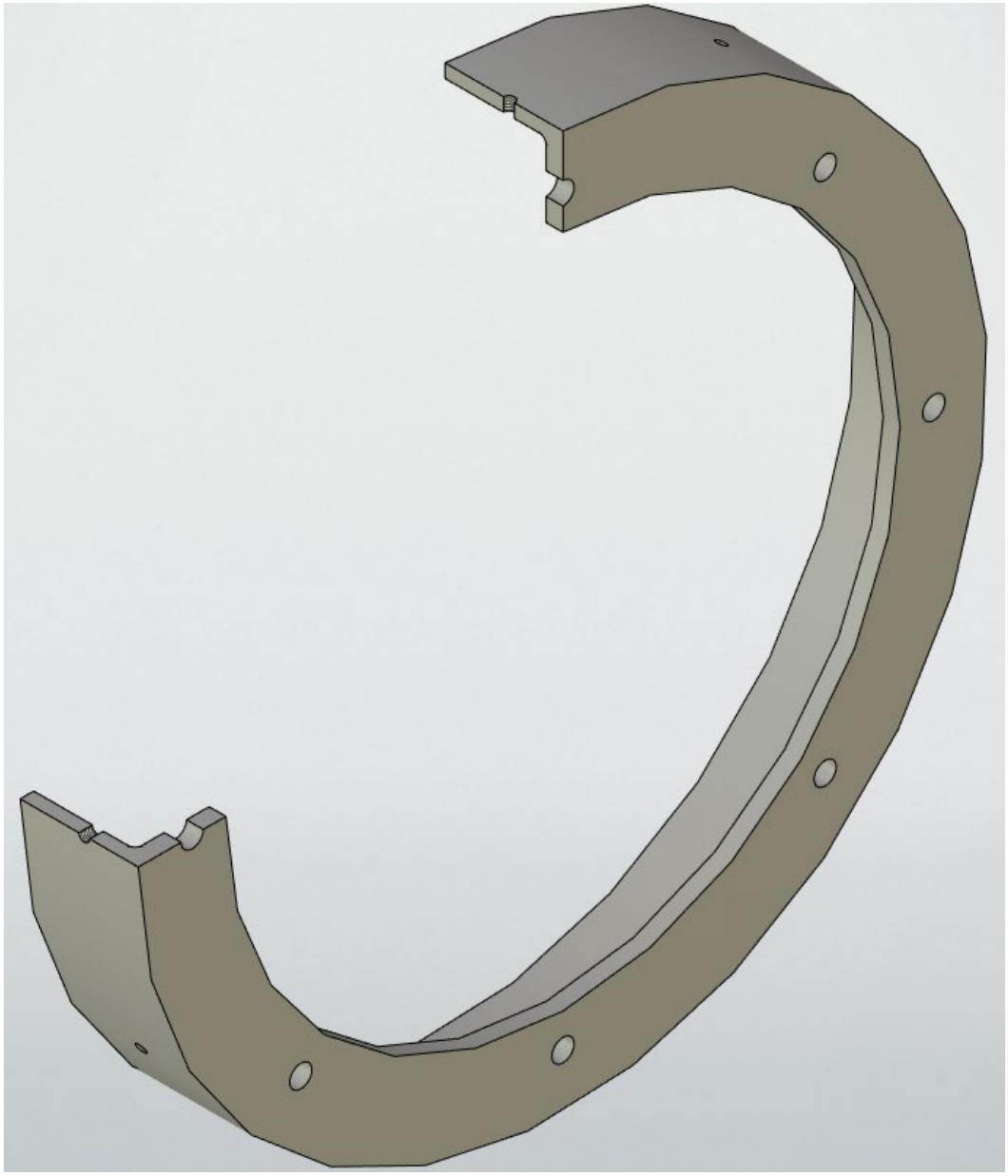
Рассечённая модель шпангоута в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



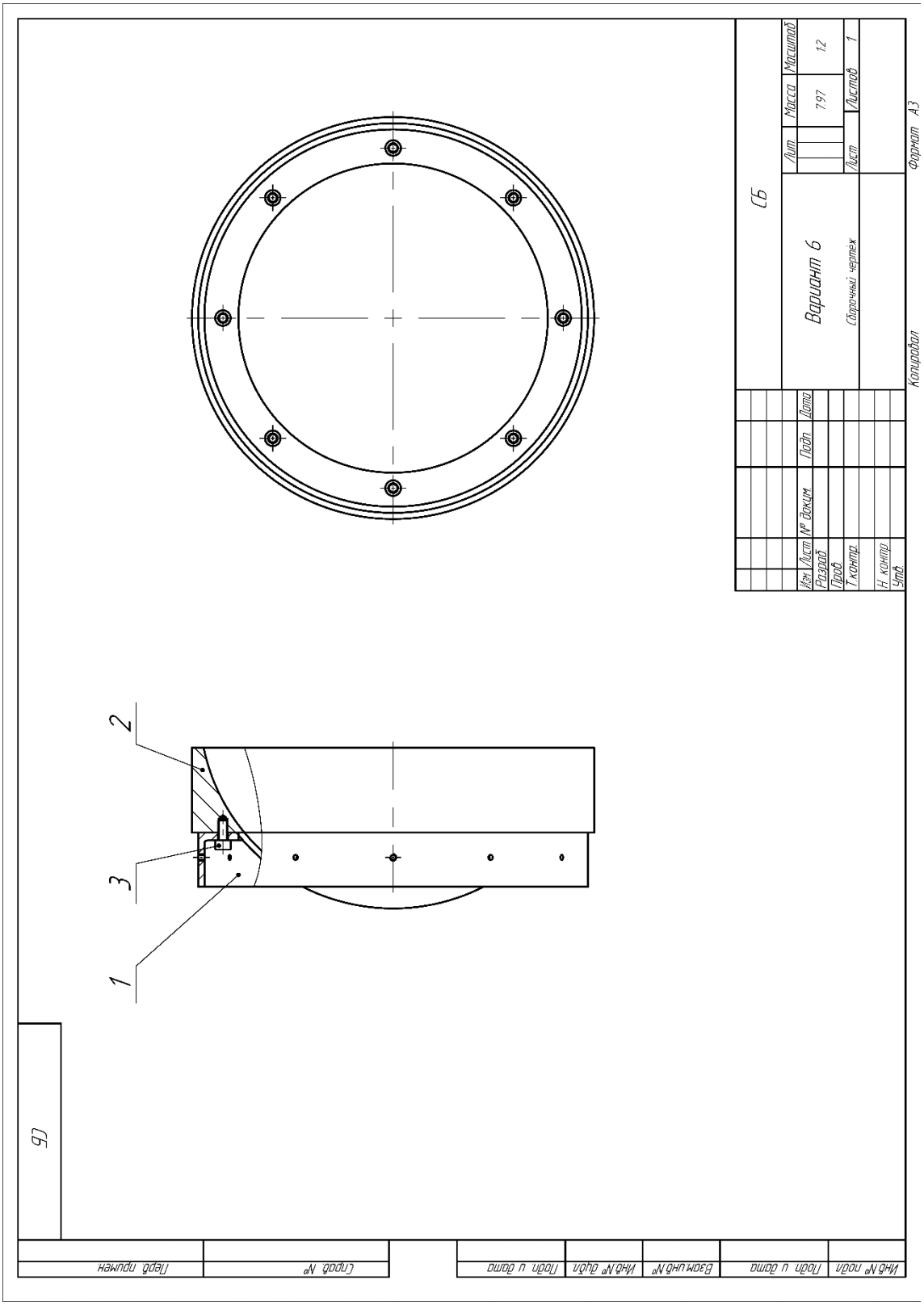
Ответ:



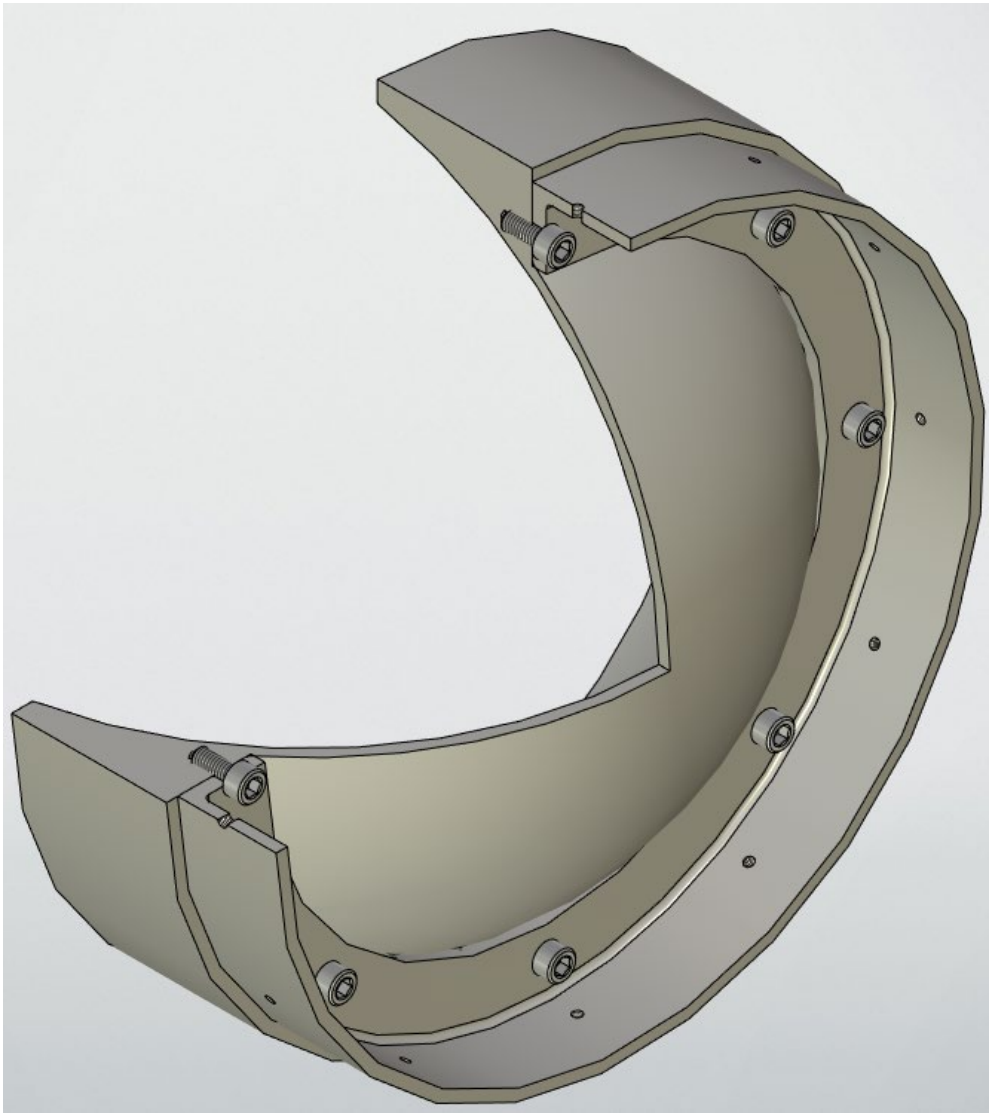
Рассечённая модель шпангоута в изометрии

Задание №3

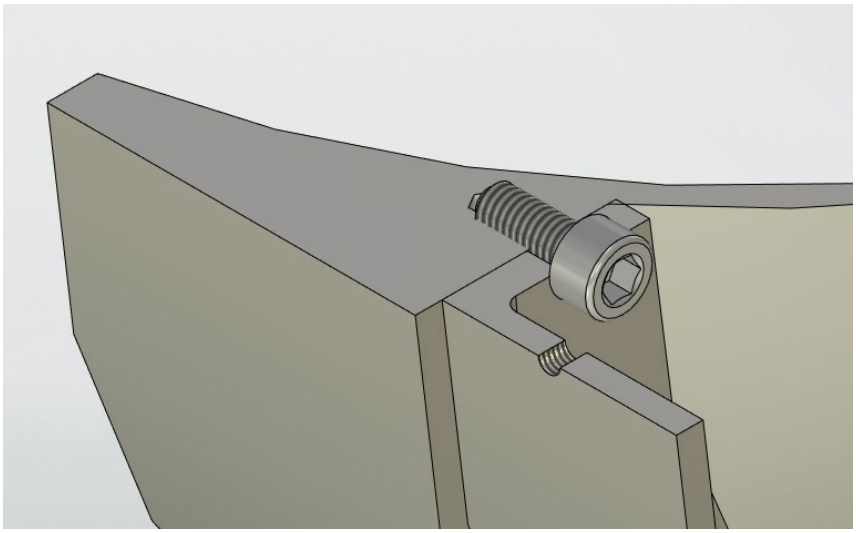
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Шпангоут	1	
А3		2		Шпангоут герметичный	1	
				Стандартные изделия		
		3		Винт М6×14 ГОСТ 11738-84	8	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 6

Лит.	Лист	Листов
		1

Копировал

Формат А4

Вариант 7

Кейс №1 «Программирование»

«Строка с максимальной суммой простых чисел»

Разработать на языке Python программу, которая находит строку матрицы с максимальной суммой простых чисел.

Пользователь вводит размеры матрицы n и m, затем n × m целых положительных чисел.

Требуется:

Сформировать матрицу n × m из введённых чисел;

Для каждой строки найти:

- Список простых чисел в строке;

Определить строку с максимальной суммой простых чисел. Если таких несколько — выбрать первую по индексу;

Вывести:

- Номер строки с максимальной суммой простых чисел;
- Простые числа в строке (отсортированные по возрастанию методом пузырьковой сортировки);
- Сумму простых чисел в строке;
- Среднее значение простых чисел в этой строке (с округлением до двух знаков);

Если в матрице нет ни одного простого числа — вывести сообщение: «Простых чисел не найдено.»

Таблица с функциями:

Функция	Параметры	Описание	Пример вызова функции
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)
bubble_sort	массив (список): **list[int]**	Сортировка пузырьком по возрастанию	bubble_sort([5, 3, 8])

find_max_prime_row	matrix: list[list[int]]	Поиск строки с наибольшей суммой простых чисел	find_max_prime_row(matrix)
print_row_info	index: int, primes: list[int]	Вывод информации о строке	print_row_info(1, [3, 5, 7])

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размеры n и m от 1 до 100.
При равной частоте выбирается наименьшее по значению простое число.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются).
Выбирается первая строка с максимальной суммой.
Функция is_prime(n) проверяет простоту через деление до √n.
Простые числа определяются вручную, без sympy и других библиотек.
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
При отсутствии подходящих чисел выводится сообщение: «Простых чисел не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:
def main():

```
n, m = map(int, input("Введите размеры матрицы: ").split())

print("Введите элементы матрицы построчно:")
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные

Пример 1: Строка с наибольшей суммой простых

Ввод:

3 4

2 3 5 7

4 6 8 10

11 13 17 4

Вывод:

Строка с максимальной суммой простых чисел: 2

Простые числа строки (отсортированы): [11, 13, 17]

Сумма: 41

Среднее значение: 13.67

Пример 2: Несколько строк с одинаковой суммой — берём первую

Ввод:

2 3

3 5 7

2 5 8

Вывод:

Строка с максимальной суммой простых чисел: 0

Простые числа строки (отсортированы): [3, 5, 7]

Сумма: 15

Среднее значение: 5.00

Пример 3: Все строки без простых чисел

Ввод:

2 2

4 6

8 9

Вывод:

Простых чисел не найдено.

Пример 4: Только одна строка содержит простые

Ввод:

3 3

4 6 8

2 3 5

9 10 12

Вывод:

Строка с максимальной суммой простых чисел: 1

Простые числа строки (отсортированы): [2, 3, 5]

Сумма: 10

Среднее значение: 3.33

Пример 5: Все строки содержат по одному простому числу

Ввод:

3 3

4 6 3

5 8 10

7 9 12

Вывод:

Строка с максимальной суммой простых чисел: 2

Простые числа строки (отсортированы): [7]

Сумма: 7

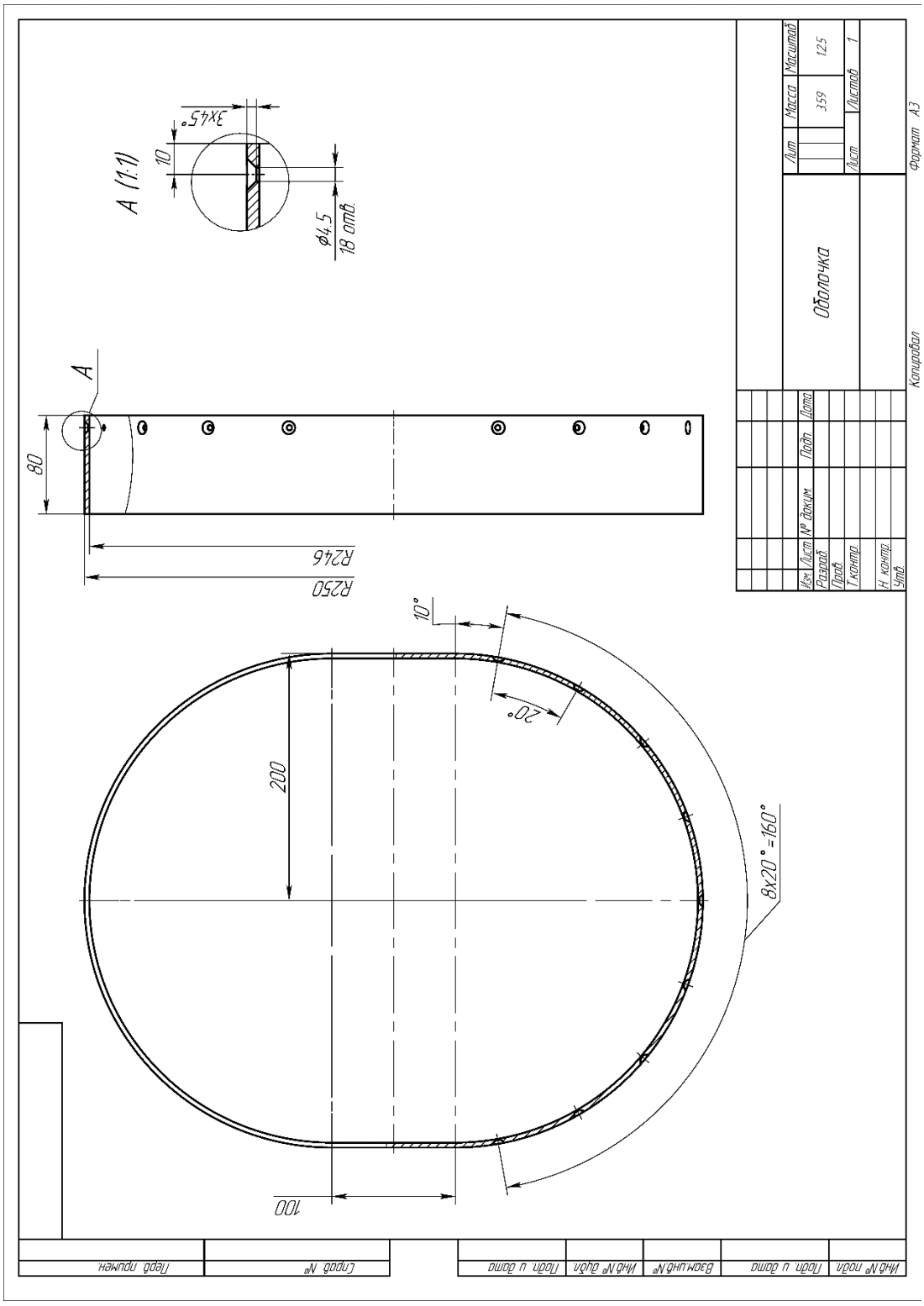
Среднее значение: 7.00

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

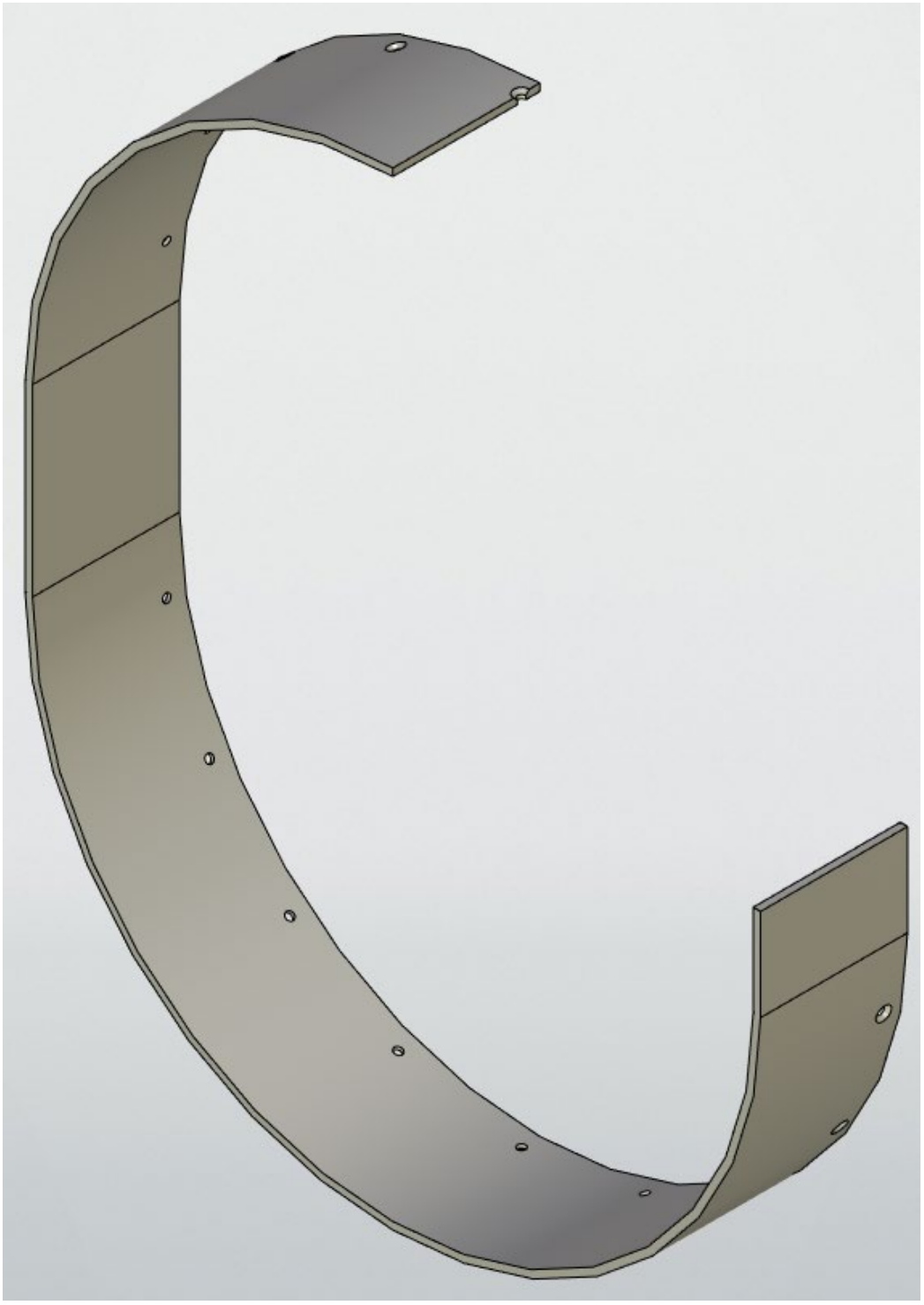
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



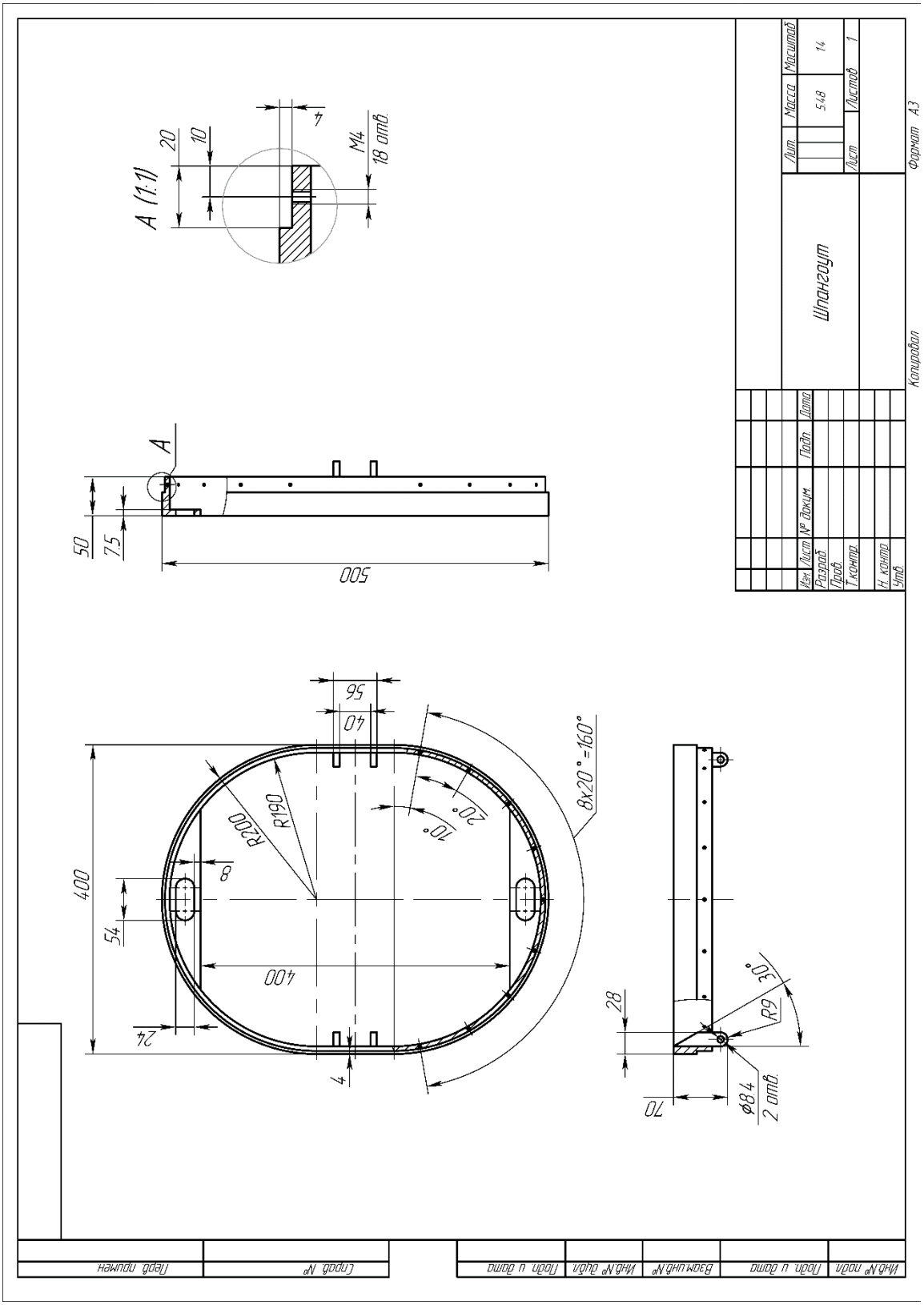
Ответ:



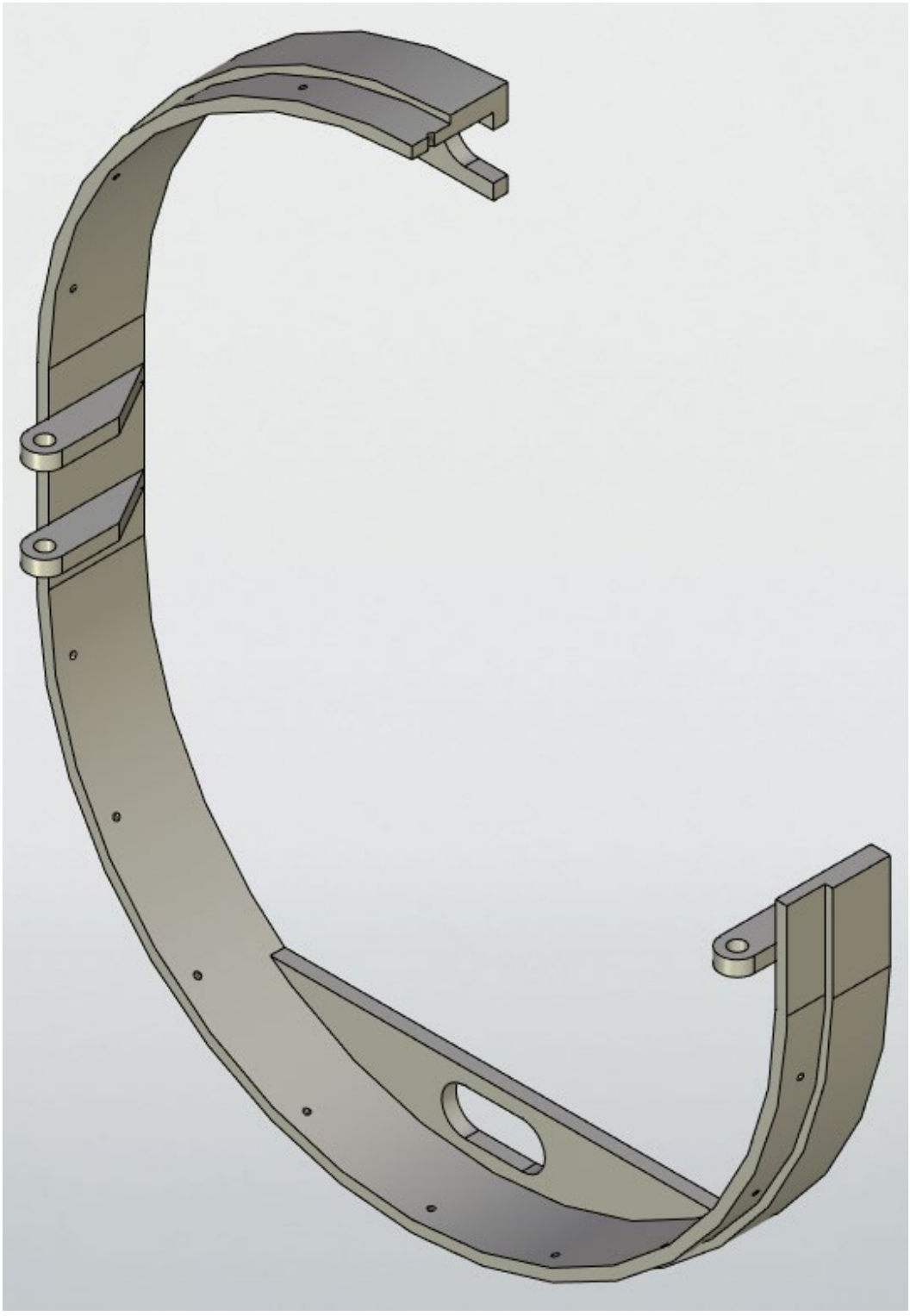
Рассечённая модель оболочки в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



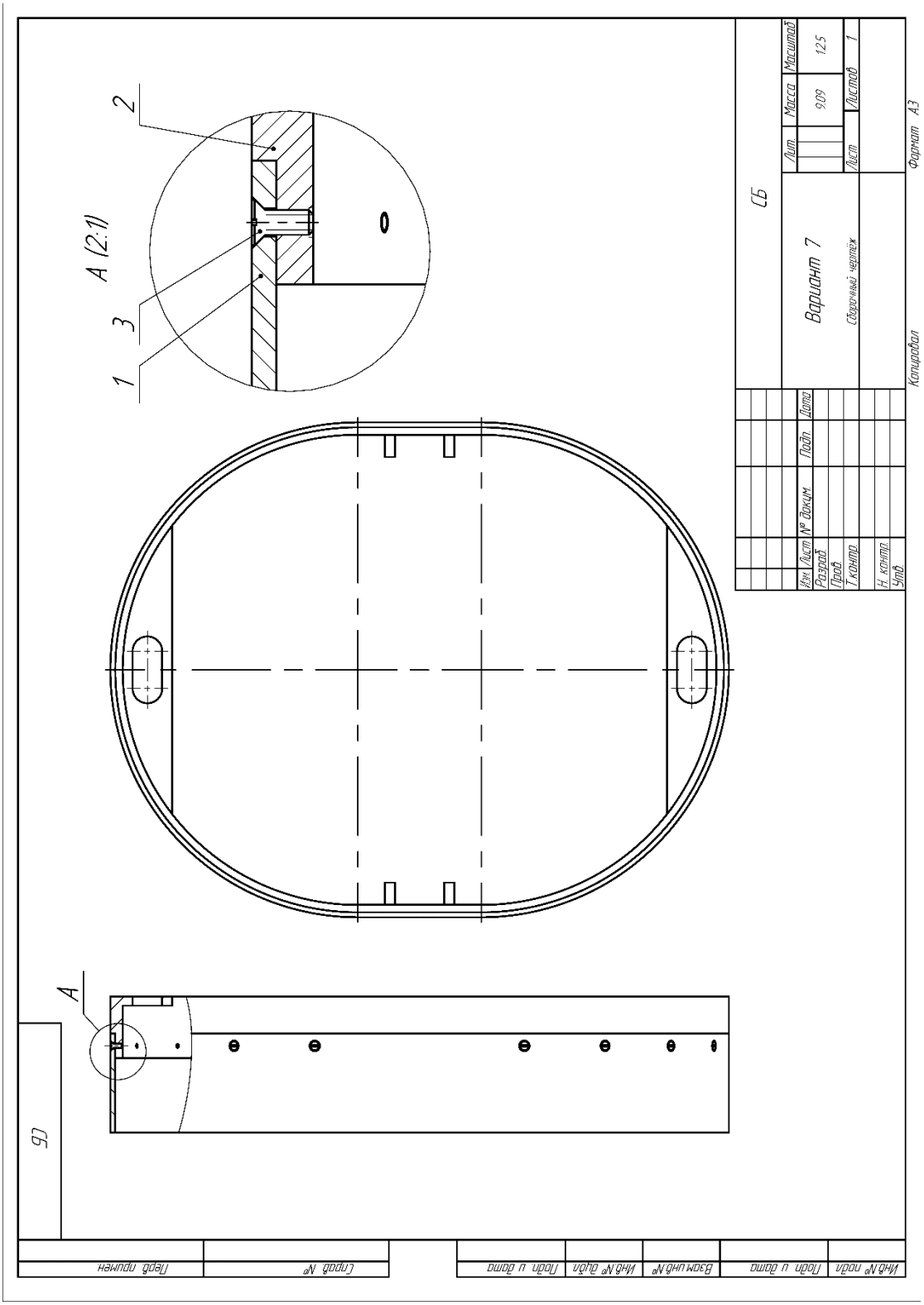
Ответ:



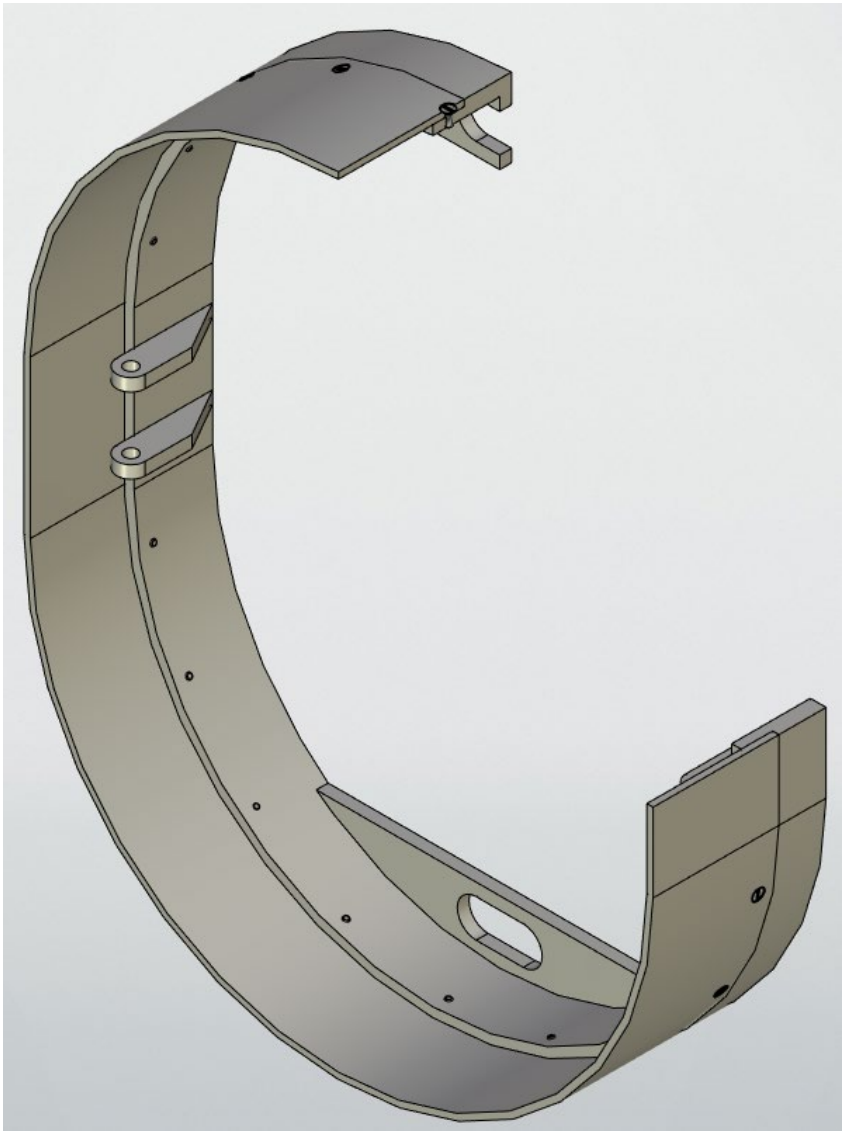
Рассечённая модель шпангоута в изометрии

Задание №3

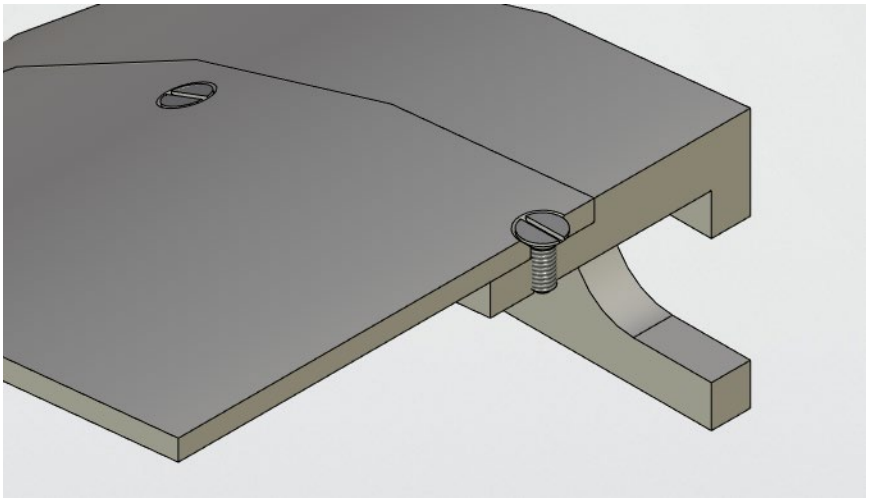
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Оболочка	1	
А3		2		Шпангоут	1	
				Стандартные изделия		
		3		Винт М4×10		
				ГОСТ 17475-80	18	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 7

Лит.	Лист	Листов
		1

Копировал

Формат А4

Вариант 8

Кейс №1 «Программирование»

«Столбец с наибольшим количеством простых чисел»

Разработать на языке Python программу, которая находит столбец с наибольшим количеством простых чисел в матрице.

Пользователь вводит размеры матрицы n и m, затем n × m целых положительных чисел. Требуется:

Сформировать матрицу n × m из введённых чисел;

Для каждого столбца:

- Посчитать количество простых чисел;
- Собрать их в список;

Найти столбец с наибольшим количеством простых чисел. Если таких несколько — выбрать первый по индексу;

Вывести:

- Номер столбца;
- Список простых чисел в этом столбце (отсортированных по возрастанию методом пузырьковой сортировки);
- Количество найденных простых чисел;
- Среднее значение простых чисел (округлённое до двух знаков);
- В выводе среднее значение рассчитывается только для простых чисел столбца

Если в матрице нет простых чисел — вывести сообщение: «Простых чисел не найдено.»

Таблица с функциями:

Функция	**Параметры**	**Описание**	**Пример вызова функции**
is_prime	число: **int**	Проверка числа на простоту	is_prime(13)

bubble_sort	массив (список): **list[int]**	Сортировка пузырьком по возрастанию	bubble_sort([5, 3, 8])
find_best_column	matrix: list[list[int]]	Поиск столбца с наибольшим числом простых чисел	find_best_column(matrix)
print_column_info	index: int, primes: list[int]	Вывод результата	print_column_info(0, [2, 3, 5])

Указания для задания:

Указание типов входных параметров в явном виде необязательно
Предполагается, что программе на вход подаются только корректные (не вызывающие ошибок) последовательности команд.
Размеры n и m от 1 до 100.
При равной частоте выбирается наименьшее по значению простое число.
Элементы матрицы - целые положительные числа (неотрицательные и ноль не используются).
Если несколько столбцов содержат одинаковое максимальное количество простых — выбирается первый.
Функция is_prime(n) проверяет простоту через деление до √n.
Простые числа определяются вручную, без sympy и других библиотек.
Функция bubble_sort(lst) сортирует вручную, без использования .sort() или sorted().
При отсутствии подходящих чисел выводится сообщение: «Простых чисел не найдено.»

Интерфейс для работы с пользователем:

Интерфейс должен быть реализован в виде текстовых команд, вводимых пользователем. Примерная структура интерфейса:
def main():

```
n, m = map(int, input("Введите размеры матрицы: ").split())

print("Введите элементы матрицы построчно:")
```

А дальше необходимо реализовать корректный ввод данных, а также вызов соответствующей функции и обработки её результата.

Ответы на разные входные данные

Пример 1: Явно доминирующий столбец

Ввод:

3 4

2 4 6 8

3 9 10 12

5 14 15 16

Вывод:

Столбец с наибольшим количеством простых чисел: 0

Простые числа столбца (отсортированы): [2, 3, 5]

Количество простых: 3

Среднее значение: 3.33

Пример 2: Все столбцы равны — берём первый

Ввод:

2 3

2 3 5

7 11 13

Вывод:

Столбец с наибольшим количеством простых чисел: 0

Простые числа столбца (отсортированы): [2, 7]

Количество простых: 2

Среднее значение: 4.50

Пример 3: Только один столбец содержит простые числа

Ввод:

3 3

4 6 8

9 7 10

11 12 14

Вывод:

Столбец с наибольшим количеством простых чисел: 1

Простые числа столбца (отсортированы): [7]

Количество простых: 1

Среднее значение: 7.00

Пример 4: В матрице нет простых чисел

Ввод:

2 2

4 6

8 10

Вывод:

Простых чисел не найдено.

Пример 5: Один единственный простой элемент

Ввод:

3 3

4 6 8

9 5 10

12 14 16

Вывод:

Столбец с наибольшим количеством простых чисел: 1

Простые числа столбца (отсортированы): [5]

Количество простых: 1

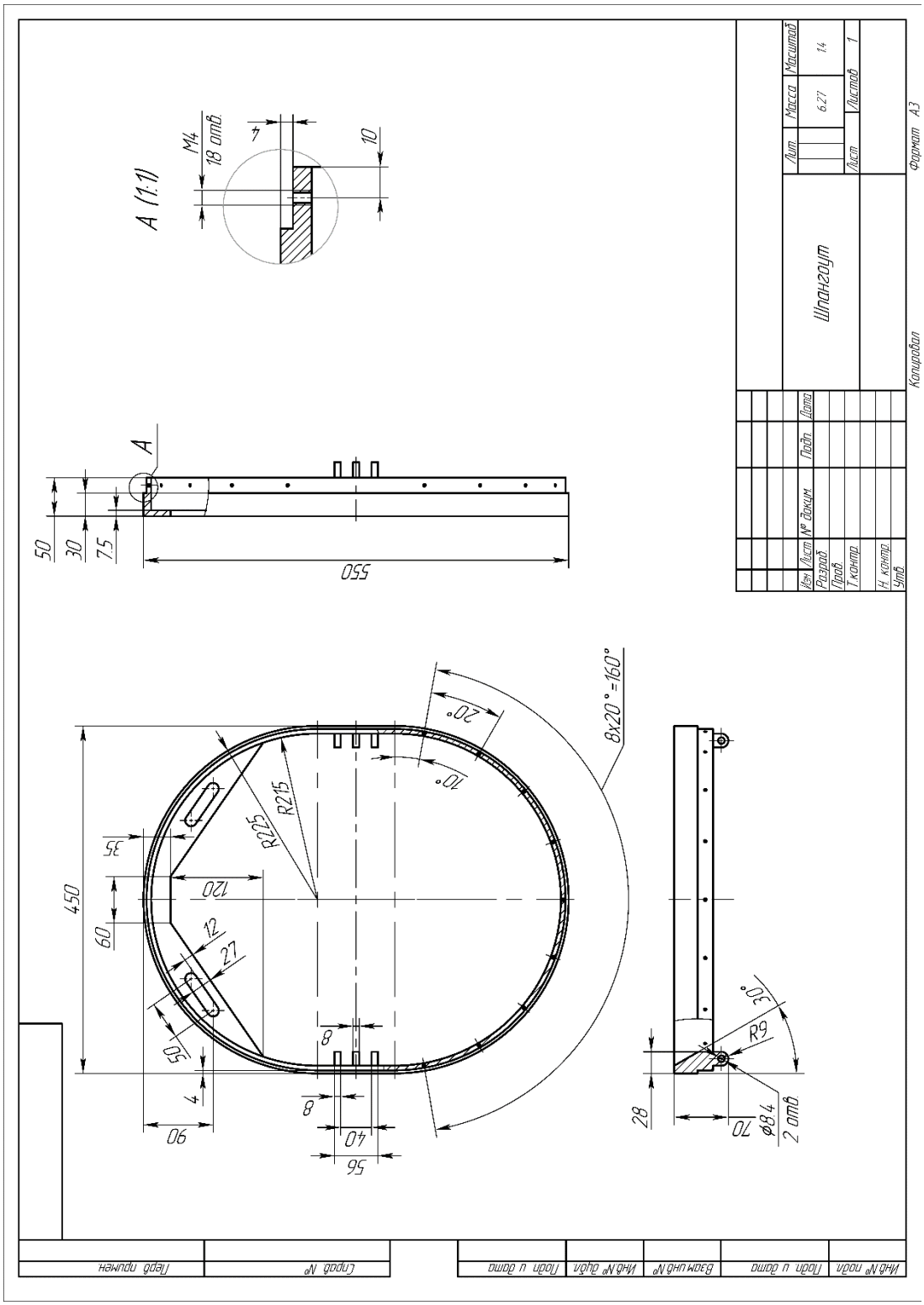
Среднее значение: 5.00

ИНТЕЛЛЕКТУАЛЬНЫЙ МЕГАПОЛИС

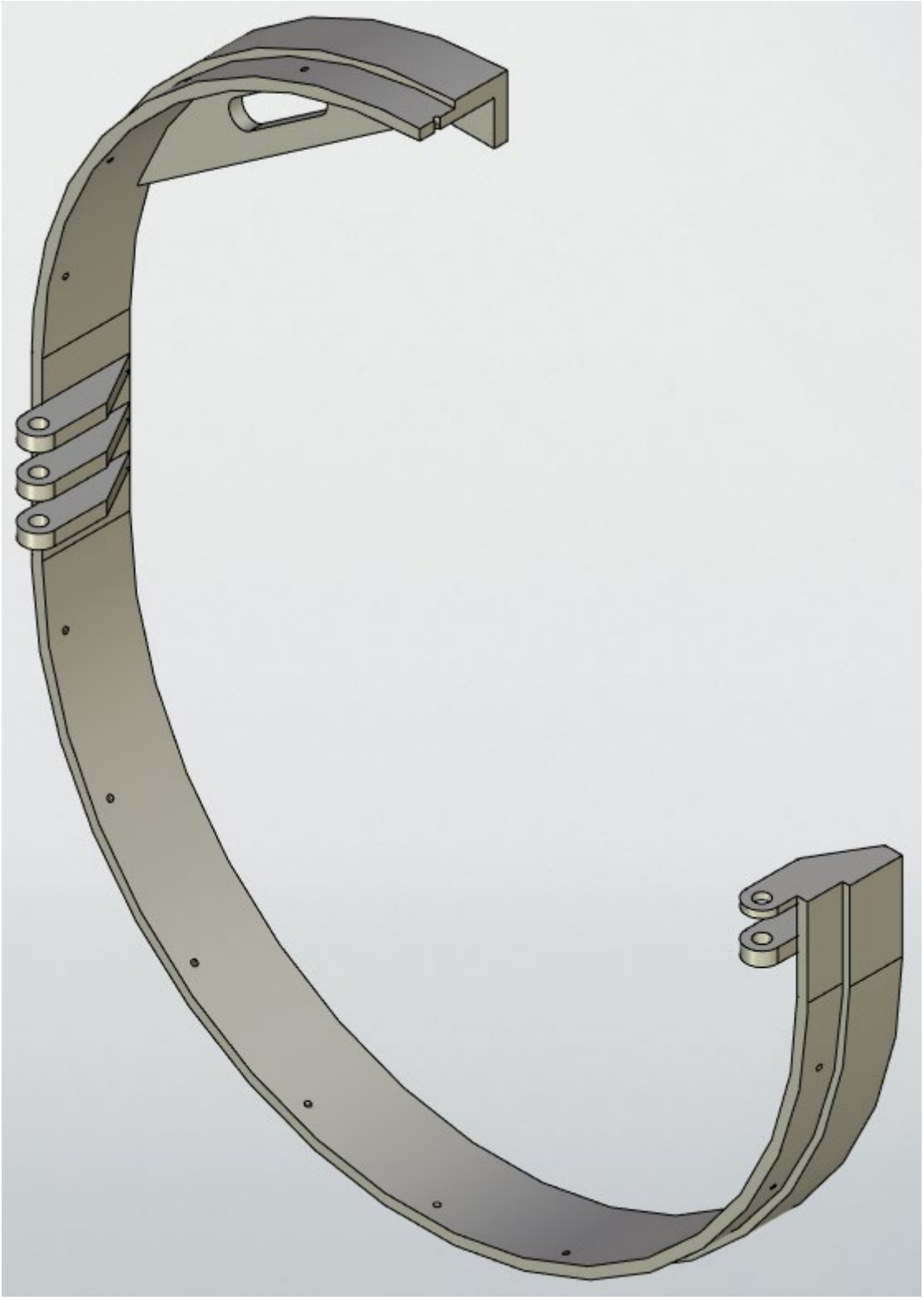
Кейс №2 «3D-моделирование и 3D-печать»

Задание №1

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



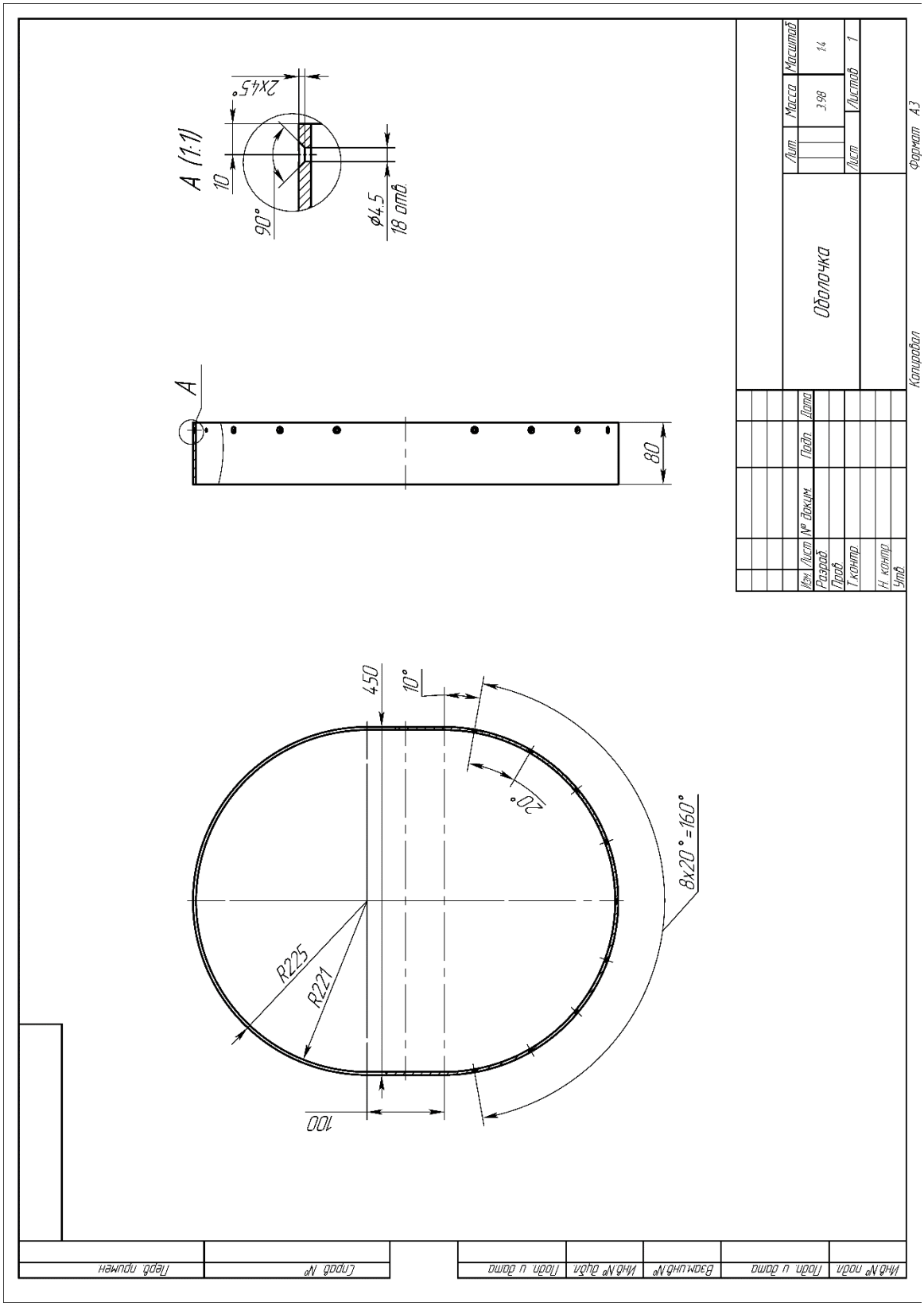
Ответ:



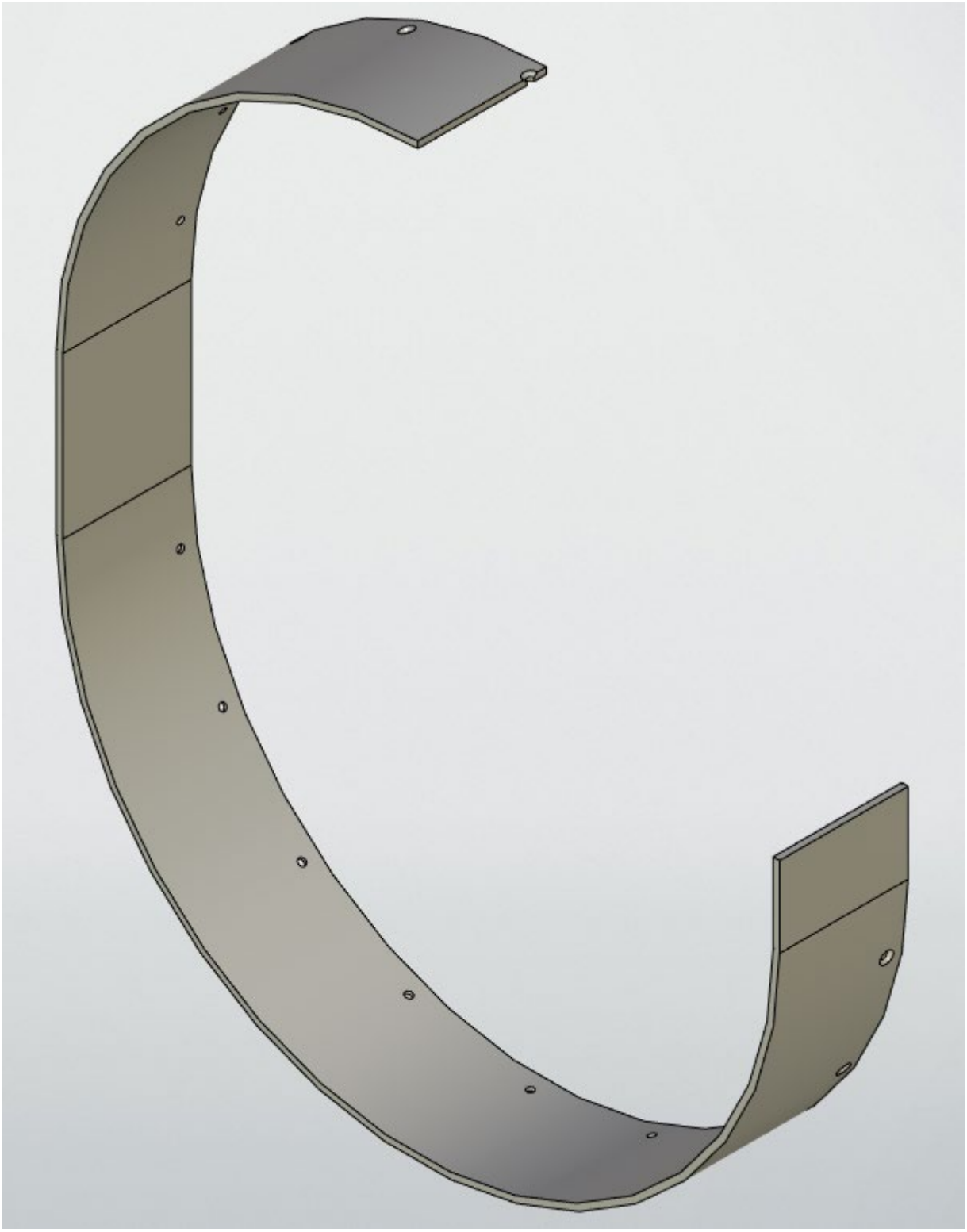
Рассечённая модель шпангоута в изометрии

Задание №2

Разработать трехмерную модель элемента летательного аппарата согласно чертежу.



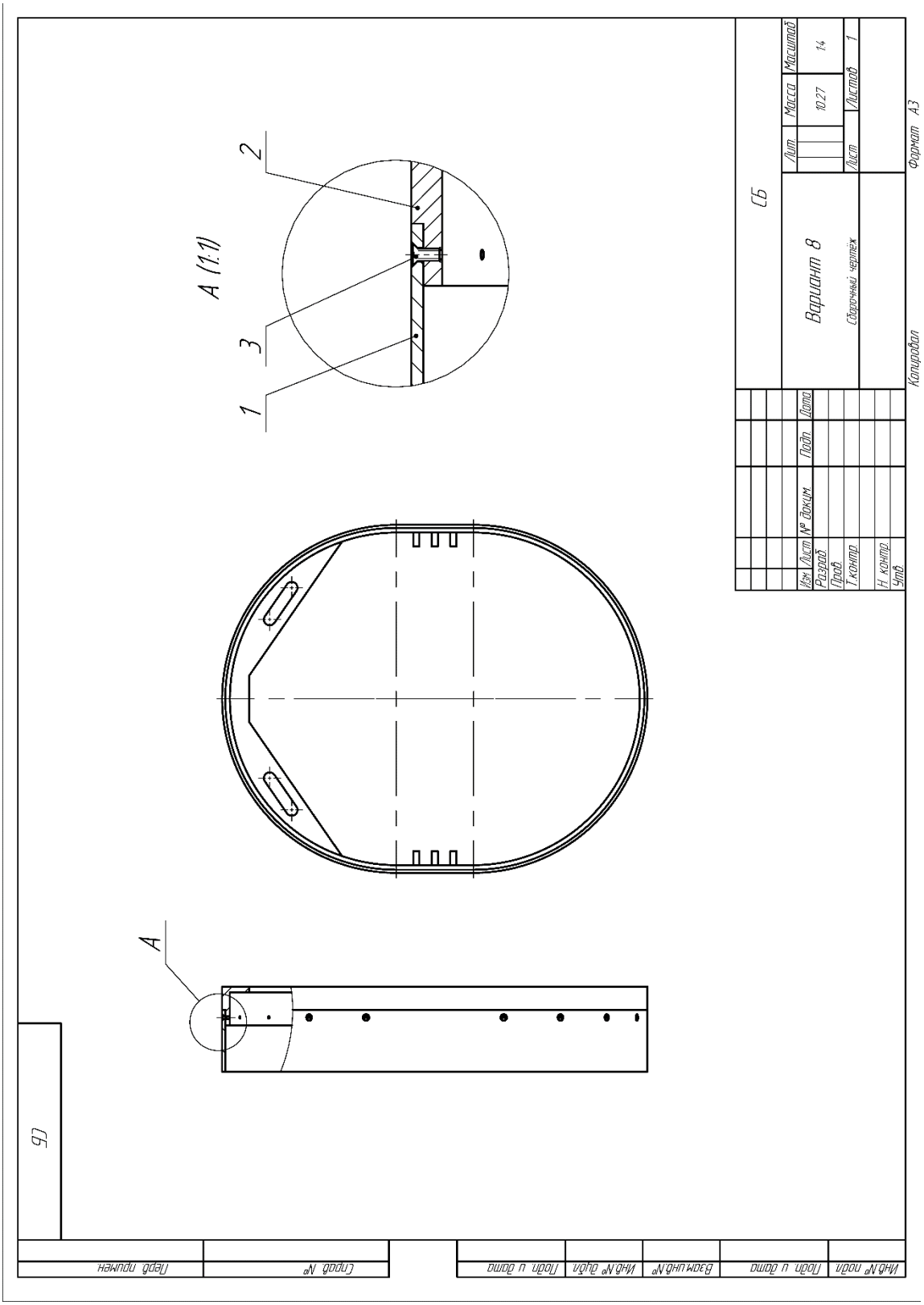
Ответ:



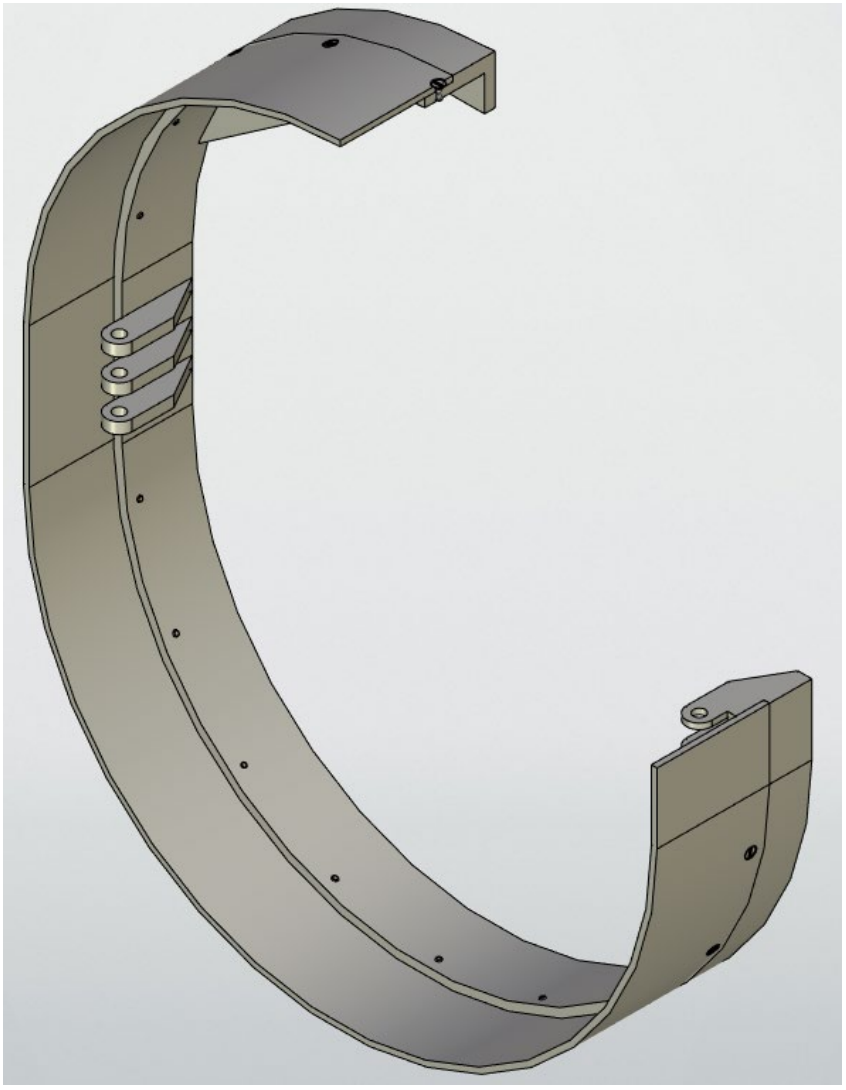
Рассечённая модель оболочки в изометрии

Задание №3

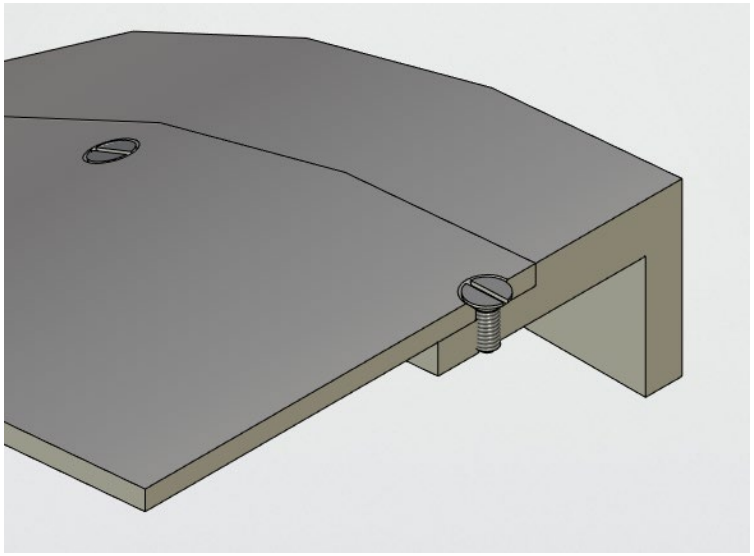
Разработать сборочную модель в соответствии со сборочным чертежом и спецификацией. Стандартные изделия должны быть добавлены из соответствующих библиотек указанном программном обеспечении.



Ответ:



Рассечённая модель сборки в изометрии



Укрупнённый вид резьбового соединения в сборке

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание
				Документация		
А3			СБ	Сборочный чертеж		
				Детали		
А3		1		Оболочка	1	
А3		2		Шпангоут	1	
				Стандартные изделия		
		3		Винт М4×10		
				ГОСТ 17475-80	18	

Перв. примен.

Справ. №

Подп. и дата

Инв.№ дубл.

Взам.инв.№

Подп. и дата

Инв.№ подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Н.контр.				
Утв.				

Вариант 8

Лит.	Лист	Листов
		1

Копировал

Формат А4