Very important points on PMI-ACP Exam: -

Understand Then Understand For Understating Then Practicing The Exam

1. You should understand the **Roles** and **Responsibilities** of Agile Practitioner -Product Owner -Scrum Master/Agile Coach/Team Lead, Development team and Sponsor.
2. Many questions as an agile practitioner what you will do in these situations.

Such as: As a Scrum Master what should be your action? -- As a Product Owner what should be your action? -- As a Development Team what should be your action?

3. Any question asking "what should have been done" for a case with faulty result, therefore considering preventive actions was necessary.
4. Agile practitioner embraces the idea of **Refactoring** which means Cleans up the code to make it: – Simpler– Easier to understand – Adhere to standards – Future work
5. As the company starting to introduce agile to its employees. It is always better to balance teaching principles with Agile practices and make them **be agile** not just **doing** it.
6. In exam there are no much mathematical questions **one or two** questions.
7. **Broken Comb** is a person who has various depths of specialization in **multiple skills** required by the team.
8. **Behavior-Driven Development (BDD**) involves a system design and validation practice that uses test-first principles and English-like scripts
9. A Business Requirement Document (**BRD**) lists all requirements for a specific project.
10. Changes may change the prioritize list in the product backlog and may bump out some priorities from the list.
11. Some questions on Prioritization (T&T) Moscow, Kano model, Dot Voting and Risk value quadrant, Also Estimation(T&T) -Planning Poker and Affinity
12. **Important terms and topics** Osmatic communication &Co-Located team, Value stream map, Kanban board, burndown chart, information radiator, interpersonal skill, servant leadership, product backlog refinement, issues, risks, project charter, WIP and Spike
13. There has to be a minimum of **three teams** to apply the Scrum of Scrums method; two base teams plus an overarching team
14. **KanBan Boards** are **pull systems** 1-They visualize the project 2- They are sign boards 3-KanBan in Japanese means "cards you can see.
15. If the customer wants you to add new high important requirements to the next iteration. Then, it is recommended to review the current planned stories and check if the **high risky with low value story** can be removed.
16. If the story is **complex** and cannot be completed within the **time-box of one iteration**, the story should be **sliced**. Whether the story should be replaced with another one, or should the developer consult with the team only or with the team and product owner
17. If there are **bugs** found later in the accepted work, these should be treated **as new stories** but it is a priority call taken by **product owner** whether to add these in current iteration or later
18. You can calculate your efficiency using (**Value Stream**)
19. Minimal Marketable Feature (MMF) can increase the ROI.
20. **Cadence** means rhythm of execution. Time-boxing is one specialized form of cadence. The time box of a sprint helps to create a regular cadence for the development team, management team, customer, etc. This rhythm helps to keep everyone in sync and keep the process moving.
21. **Technical debt** is a major problem in may products that is generally because of poor agile practices followed.

Agile practices focus on the reduction of technical debt and there are three main proactive practices that can be followed for the same purposes. These practices increase technical excellence with good code design by implementing refactoring and using automated testing tools. Including technical debt reduction user stories in the backlog is a reactive methodology. In this, technical debt is already incurred in user stories and user stories have been completed for eliminating the debt. Therefore, it is the least likely approach that you should recommend to improve the quality of the upcoming projects as an agile practitioner.

22. You are selecting interpersonal skill from the following ones to use. Within your team **(Communication or Leadership).** Best answer is **communication** as sometimes they define Interpersonal skills as communication between two or more people
23. **Refactoring:** A change that is made to the internal structure of software to make it easier to understand and cheaper to modify without changing its observable behavior ,Also is changing code, without changing its functionality, to make it more maintainable, easier to read, easier to test, or easier to extend. Also is a technique for restructuring an existing body of code by improving/simplifying its internal structure (design) without changing its external behavior. Also, is one of the principal techniques for managing technical debt. (A technique to improve external functionality.)
24. **Technical debt** can arise during initial development, ongoing maintenance (keeping a product at its original state), or enhancement (adding functionality).(All Times)
25. During sprint If there is **potential problem or risk** usually contact Product Owner.
26. During Sprint If there is **Significant /essential or major risk**, contact Stakeholder
27. Question about: **disagreement on the requirement**, someone say it is there and another say it is not there and it is not meeting the requirement **(Definition of Done)**
28. While you are presenting the demo, it was not accepted as it is not meeting the requirements **(acceptance Criteria)**
29. When new stakeholders engage the project, the agile practitioner need to **learn how** the current project may be affected by these stakeholders.
30. **A Fishbowl window** is a long-lived video conferencing link between various locations. People start the link at the beginning of a workday and close it at the end. In this way, people can see and engage spontaneously with each other, reducing the collaboration lag otherwise inherent in the geographical separation. An effective communication tool may reduce the risk of scope creep but doesn't reduce the project scope. This might increase the direct project costs (due to the use of technology) but, at the same time, might reduce the indirect project costs (due to efficient WIP management).
31. The document in Agile are **barely sufficient**. (In Agile writing heavily, documentations are not the right way to deal with challenges).
32. **The team augmentation arrangement** is regarded as the most **collaborative contracting approach** as this embeds the supplier's services directly into the customer organization. Funding teams instead of a specific scope preserves the customer's strategic discretion on what work should actually be done.
33. Your team made an innovative way to implement the solution in a new way. Should the team directly implement it or check it with the product owner

- One case team should directly implement it as team is self-organized and self-directed
- Another case you will need to return back to the case will clear it Fully

34. In case when there are issues/risks/dependencies/ changes. After describing the case ask you for the solution and you see in the answer (Describe or explain to the customer working software is better than compressive documentation or other agile values) **Usually those are Trap questions** make sure you understand it very good as the answer in many case is not that direct choice. Sometimes implement the solution or discuss the solution with the customer or the product owner before you implement it.

35. If senior manager is asking for many reports –Educate him that the agile working software is over comprehensive documentation **OR** Refer him to the information radiator.
36. When you read the following words / phrases, pay special attention as the answer choices may not be correct: **change management-create a plan / amend a plan-assign tasks/scope creep-escalation-report and documentation-baseline-Monte Carlo, regression analysis, etc.**

37. The following words will guide you to eliminate this option such as:

assigned (tasks / user stories) – **escalate** (the situation) – the very nature of the Agile team formation (with Product Owner / Customer co-located) makes escalation of issues much less than traditional project management **plan** – Agile projects seldom create lots of project management plan upfront- **documentatio**n – though Agile highly favors working software, it is perfectly okay to create documentations that are **barely sufficient**, don't rule out creating documentation as a possible task- **direct, instruct, request, Force, push** (by the project leader) the leader is often a servant leader and would seldom assume a "command and control" position

38. The following words will guide you to select this option such as (**Motivate, Assist, Self-organized, high value, in last responsible moment, guide, support, help, Innovate** etc.)
39. If team member came with innovative idea and ask for permission to make it. Then ask you what you will do: **Support him & let him try it in the next iteration.**

40. If team member is overloaded and is thinking that the work has no end. (**To give him a break** or to give him vacation)
41. Product Owner is not needed to attend sprint regular meetings. Only some of them. The retrospective meeting is used to check what went well in the last sprint and enable improving for next iteration also it is an opportunity for the team to inspect itself and create a plan for improvements to be enacted during the next sprint.
42. As agile coach you reward and recognize innovative failure. Its ok to fail as long as team learn and improve.

43. **Agile**, Encouraging the **generalizing specialist**, not overtime to avoid lack of resources.
44. If there are too many interruptions to the team, you need as an agile practitioner to work toward shield team from distractions.

45. Agile practitioner is coaching the team, and support them to be more self- organize and self-direct. Failing fast will help the team to learn the right way

46. **Reciprocal Commitment**, where the Agile project team **commits** to delivering the specified functionality

100 % according the **definition of done** at the end of the iteration, and the product owner, organization and customer agree not to **change priorities** during the iteration and leave the team alone to "do the work".
47. We communicate in two ways, **Synchronously and Asynchronously**.
Synchronous communication is when people are focused continuously until the communication is finished. **Asynchronous communication** is using a pattern that allows for focusing on something else while waiting for the other person to formulate a response. Increasing asynchronous communication is a good place to start.

48. **User story vs Task** – user stories will add value (bearing story points) but that is not necessarily true for tasks

49. **Affinity estimating vs relative sizing** – both have similar meaning, relative sizing is estimating the efforts in relation to other user stories while affinity estimating is the use of common sizing units to compare task sizes (like S,M,L,XL or coffee cup sizes), affinity estimating is usually used for release planning (**early stage**) while relative sizing is used in iteration planning

50. **sprint review vs sprint retrospective** – sprint review is for demo and validation purposes, often involving customer and other stakeholders; sprint retrospective is for the team members only with the purpose of finding areas for improvements (including people, process and product)

51. **A user story workshop** is a meeting that includes developers, users, the product customer, and key stakeholders who all collaborate to write user stories
52. **Parkinson's Law** states "Work expands so as to fill the time available for its completion." In Agile processes, we use the notion of a "timebox" or "spike" to deal with this sort of work.
53. **Exploratory testing**: Testing is used to look for surprises and gaps in the software and provides an after-the-fact check on the team's practices

54. **Hardening iteration (H):** Many agile teams will set aside an iteration at the end of the project to use for "hardening" (preparation for final rollout of the product). Note that this is not a bug-fixing iteration! For an internal release, this hardening iteration may consist of a variety of production-readiness activities, whereas for an external release, additional tasks such as capturing final screenshots for promotional materials and lining up production facilities might be items to consider.
55. **Iteration Zero, or sprint zero**, allows a team that is perhaps **newer** to Agile and doesn't have a process intact yet to develop their process, or if a **seasoned** team determines that they need a discovery iteration to develop the best approach, they would use Iteration Zero.

56. **Osmotic communication**: When osmotic communication is in place, questions and answers flow naturally and with surprisingly little disturbance among the team.
57. **Extrinsic and intrinsic quality** Customer quality (which is extrinsic or external) delivers value in the short term. Technical quality (which is intrinsic or internal) enables continuous delivery of value over time. Poor quality work results in unreliable products, but more critically, it results in products that are far less responsive to future customer needs.
58. **Stories and Use cases:** One of the most obvious differences between **stories** and **use cases** is their scope. Both are sized to deliver business value, but stories are kept smaller in scope because we place constraints on their size (such as no more than ten days of development work) so that they may be used in scheduling work. **A use case** almost always covers a much larger scope than a story.

59. User stories or other items that are likely to be more distant than a few iterations can be left as **Epics** or **themes**. That's why they are primally located at the bottom. **Epic** is often called Capability.
60. **Theme**: A set of related user stories may be combined together (usually by a paper clip if working with note cards) and treated as a single entity for either estimating or release planning. Such a set of user stories is referred to as a **theme**
61. Where the **user stories** of a release plan are estimated in story points or ideal days, the tasks on the iteration plan are estimated in ideal hours.

62. **Iterations** are **timeboxed**, so schedule and budget are always fixed too. During iteration planning we talk about the tasks that will be needed to transform a feature request into working and tested software
63. **Feature overrun**: Frequently a feature will surprise you when you start developing it. The code takes longer than expected, or you underestimated complexity. Here are some of the ways teams adapt to feature overrun: Work with the customer to prioritize the functionality in the feature and potentially reduce the scope, Accept the discovery and continue the work into the next iteration, Cancel the feature, and re-evaluate the feasibility of the project

64. **Definition of Done**: When the Product Backlog item or an Increment is described as "**Done**", everyone must understand what "Done" means. Although this varies significantly per Scrum Team, members must have a shared understanding of what it means for work to be complete, to ensure transparency.
    This is the "**Definition of Done"** for the Scrum Team and is used to assess when work is complete on the product Increment. The same definition guides the Development Team in knowing how many Product Backlog items it can select during a Sprint Planning Meeting. Development Teams deliver an Increment of product functionality every Sprint. This Increment is useable, so a Product Owner may choose to immediately release it. The quality criteria are captured in the definition of done.

65. **A Swarm** is a temporary group of team members that works together on one story to bring it to completion.

66. **Sprint Goal**: output from a **Sprint Planning Meeting** provides the Development Team with a target and overarching direction for the Sprint
67. **Risks** are identified in all planning meetings: release, iteration, and daily stand-ups,
    Risks can be analyzed and addressed in all planning meetings, with the focus on qualitative analysis, not quantitative.
68. **Mitigate** — Take steps before the risk materializes to reduce the eventual containment costs. For example, moving a feature to an earlier iteration to ensure that it's completed
69. **Exposure** --A specific amount of time or money used for the sole purpose of absorbing risk is called Risk Exposure.
70. **Advantages of user stories**: Some of the advantages of user stories over alternative approaches are: User stories emphasize verbal communication, User stories are comprehensible by everyone, User stories are the right size for planning, User stories work for iterative development, User stories encourage deferring detail, User stories support opportunistic design, User stories encourage participatory design, User stories build up tacit knowledge.

- **User stories** originated as part of Extreme Programming. Naturally, stories fit perfectly with the other practices of Extreme Programming. However, stories also work well as the requirements approach for other processes.
- **User stories** may and should be supplemented with whatever other written information helps provide clarity against what is desired.
- **User stories** that will be worked on in the near future (the next few iterations) need to be small enough that they can be completed in a single iteration. These items should be estimated within one order of magnitude
- When a story (possibly an epic) is disaggregated into its constituent stories, the sum of the estimates for the individual stories does not need to equal the estimate of the initial story or epic.
- The front of a story card contains the story and notes about open questions while the back of the card contains details about the story in the form of tests that will prove whether or not it works as expected.

71. **Financial Parameter:** Internal Rate of Return (**IRR**) is used to express the return on project in % terms when comparing two different cash flow streams.

ROI best describe how much money you hope the project will return

NPV is most information necessary to make a final decision
ROTI: Return on Time Invested (ROTI) is used to measure the effectiveness of the retrospective meetings from the team members' perspective.

Agile Charts

Burn up charts

1. Burn up charts allow the team to track performance visually while also tracking changes in the scope of work that could affect their velocity or completion of stories in an iteration

2. Burn up charts can be converted to cumulative flow diagrams by the addition of WIP
3. Burn up charts that track total scope (rather than burn down charts) separate out the rate of progress from the scope fluctuations
4. The main reason agile teams use burn charts is to show their progress and make it easy to see the expected completion date based on the current velocity trend.
5. is not true is “Burn down charts indicate whether rate of effort changes are due to changes in progress rates or scope
6. A burnup chart shows the accumulation of tested and working feature over time.

Burn Down charts

1. Bottom of the bar is lowered when additional work is added. Bottom of the bar is raised when scope is removed. Top of the bar will be lowered when work is delivered.
2. Progress of the iteration can be determined from this graph line. A flat line means that either the team is not updating its remaining work hours, or there is an obstacle preventing progress. An upward spike means that new work was discovered

Risk burndown graph

- The best tool for monitoring risk reduction
- this shows the project's evolving risk profile in one easy-to-understand chart. The team could also use risk- based spikes for their risk reduction efforts, and their latest risk-adjusted backlog would reflect the currently expected monetary value of the remaining risks.
- Risk burn down graphs do not show the impacts of the risks on the schedule or budget, but they do show the cumulative risk severities over time. Tracking just the probabilities over time would be of little use without knowing the impacts of these risks. After all, they could all be trivial, and in that case, why would we need to be concerned

Cumulative Flow Diagram (CFD)

1. Cumulative flow diagrams help tracking and forecasting the delivery of value
2. Reveals the total in progress and completed work
3. A cumulative flow diagram allows the team to track the work in progress, the work performed over time, and cycle time. It also provides a means to interpret bottlenecks in process steps
4. To visually analyze their work in progress for each step in a process in order to determine quickly where bottlenecks exist in the overall process. What tool would help them the most?

Kanban Board

- A whiteboard with columns for various stages of work, on which you post sticky notes that represent the tasks you are working on.
- A Kanban board will show user stories progress through a series of phases that being with requirements, followed by development, and finally completed by testing.
- Kanban teams use their WIP limits to control the number of items that can be in a given stage of work at a time.
- Kanban board shows work in progress (WIP), which represents work started but not completed. Therefore, the WIP should be limited and carefully managed to maximize performance. More WIP does not equal more output; in fact, it is quite often the opposite. Also, WIP is any work that is in progress, regardless of what stage the work is at,

- Kanban teams typically use cumulative flow diagrams to visualize the flow of work through the process. This allows them to get a visual sense of the average arrival rate (how frequently work items are added), lead time (the amount of time between when a work item is requested and when it's delivered), and work in progress (the number of work items in the process at any time).
- When teams use a kanban board to visualize their workflow, they use columns to represent workflow steps, and typically use sticky notes or index cards to show individual work items flowing through the process.
  If items tend to accumulate in one column, it tells the team that step is a potential root cause for the process flow slowing down. The way to fix it is to work with the stakeholders to impose a work in progress (WIP) limit, usually by writing the maximum allowable number of work items for that step.
- In a pull system, the later step in a process pulls work from the previous one. This keeps the workload even and is the least wasteful way to get work from the beginning to the end of a process.
- The order of the practices in Kanban is: visualize workflow, limit WIP, manage flow, make process policies explicit, implement feedback loops, improve collaboratively, evolve experimentally
- Why Kanban may be considered controversial: Kanban does reduce planning and may eliminate estimation

Characteristics of Kanban:

- A) Kanban focuses on delivering value B) Kanban relies on staff specialization in each phase
- C) Kanban includes reflection as part of its process D) Kanban is as adaptable as other agile methods
- E) one limitation of Kanban, which is in conflict with the best agile practices, is its reliance on staff specialization in each phase, while agile preaches for generalizing specialists. Some argue that this may be considered a reason why Kanban is not considered agile ‘enough’.

WIP: Q1: What is the main purpose of imposing limits on working progress?

A. To optimize throughput B. To minimize resource allocation
C. To visualize lead time D. To balance workflow

For the exam, it's important to understand that the purpose of using WIP limits is to optimize throughput (speed of workflow). Although WIP limits don't focus on maximizing resource allocation (keeping everyone busy), they certainly don't aim to minimize it, either. Finally, although a Kan ban board can help a team Visualize lead time and balance workflow, that isn't the purpose of implementing WIP limits.

Q2: Your team has high levels of work in progress, and you are explaining to them why that is a problem. What isn't one of the issues you mention?

A. It hides efficiency and throughput issues. **B**. It confuses the team members' roles.
C. It doesn't deliver any return. D. It increases risk and potential rework.

To answer this question, we have to understand how work in progress impacts an agile project. If we have a lot of partially done work, how can we effectively measure efficiency and throughput? Also, ail that work
"on hold" is just sitting there, not delivering any return (value). Incomplete work also increases risk, since we might have to redo it if something changes before it is finished and accepted by the customer. The only problem listed here that isn't a result of high levels of WIP is confusing the

team's roles, so that is the correct answer

Task Board

1. Story or Task Board is actively used by the team to indicate overall status and convey the current commitment of each person.
2. A task board is an information radiator that serves the dual purpose of giving your agile project team a convenient mechanism for organizing their work, and a way to see at a glance how much work is left in the iteration.

Product RoadMap

- Creating a product roadmap is a release planning technique. A roadmap is not a list of screens and reports, because we can build a roadmap for a product even if the project has no screens and reports.
- A view of release candidates
- high-level representation of the features or themes that are to be delivered in each release

Value stream analysis include:

- step in value stream analysis is “Create a value stream map of the current process, identifying steps, queues,
  delays, and information flows.”
- Value Stream Mapping helps teams to see processes and process interactions. It allows teams to optimize the process by eliminating the waste
- Value stream analysis is a very valuable tool for detecting waste, especially waste that is caused by waiting for other teams.
- Value stream maps are a way to represent current processes visually and look for better ways to streamline those processes.

Information Radiator

- When teams encounter risks, issues, and threats to the project, an important priority should always be to communicate the status of those issues. An information radiator is a very good tool for doing that
- An agile practitioner should focus on creating information radiators so that teams have access to all of the data about their work and can make decisions about how to keep their work on track on their own.
- Information Radiator Examples: * Release plan * Automated Build Report
  * Displaying Work Breakdown *Displaying Project Status

A **decision spectrum** is a participatory decision model that teams use to engage the stakeholders and to help stakeholders contribute to the decision-making process. This tool is the only choice listed that would help the team to prioritize the issues that should be addressed in the next iteration.

Wireframes and prototypes help teams to: Confirm design

- A wireframe is a low-fidelity mock-up of what the user story represents or what success looks like

Velocity : is used for all:

A. Team’s work capacity B. Estimating work per iterationC. Maintaining list of features

- The number of story points completed in a sprint. This number is averaged over time to predict the amount of work that can be accomplished across multiple sprints

- Velocity is a historical measure. Teams use it to understand their capacity based on past performance.
- Velocity is a very effective way to use the team's actual performance from past sprints to understand their actual capacity for doing work, and using that information to forecast how much work they can accomplish in future iterations. Teams do this by assigning a relative size—typically using made-up units like story points—to each story, feature, requirement, or other item being worked on, and using the number of points per iteration to calculate the team's capacity.
- Velocity represents the rate of completion rather than the time or effort expended. Therefore, incomplete user stories are excluded from velocity estimates
- A team's velocity measures the actual work done (and therefore the team's capacity). Therefore, this metric includes all the interruptions and anything else that has occurred on the project, including the meetings, part-time resources, and scope changes that are common on projects.

Story Map

- A story map gives your team a way to collaborate with each other and create a visual release plan by organizing stories into releases. This helps your team provide forecasts for future releases to your stakeholders. And it does it at a level of detail that gives them enough information to plan effectively, without including specific details that the team can't possibly know or honestly commit to this early on.

Business value delivered chart.

The entire enterprise (business, management, and development teams) needs the line of sight to velocity (points/time) dashboard-type view of work management which in other terms is a business value delivered chart

Affinity estimating

- Assignment of user stories to various T-shirt sizes (small, medium, large, extra-large).

- Agile Estimating. Using buckets or coffee cups for estimating is similar to affinity estimating

The S-curve

- is a tool commonly used to track project spending over time. Its name is derived from the shape of the curve, which is the typical shape of project expenditures over time
- tools to use for tracking project performance

Important Roles and Responsibilities in Agile

Project Sponsor

1. Serves as the project's main advocate within the organization.
2. Provides direction to the product owner role (the person or team representing the business) about the organization's
   overall goals for the project.
3. Focuses on the big picture of whether the project will deliver the expected value on time and on budget.
4. Is invited to the iteration review meetings to see the product increments as they are completed, but might not attend.

Product Owner Responsibility

1. The product owner has the primary responsibility for ensuring that value is delivered by a Scrum project. They

do this by managing the product backlog and making sure the team has a correct understanding of the project vision, the project goals, and the product requirements.
2. The product owner has responsibility of updating the product roadmap primarily rests
3. to help the team understand customer requirements, not the Scrum Master's
4. One of the main roles of the PO is to accept the product coming from the sprint or rejecting it
5. The Sprint planning meeting cannot happen without the PO, so it should be rescheduled.
6. Only product owner may cancel the sprint
7. Product owner needs to agree on any changes on the sprint or the backlog.

Backlog grooming or refinement is the act of the product owner sorting the backlog by priority.

8. New features, risks associated with these new features or other technical risks, defects to be fixed, are among the different factors that the product owner has to consider when adding items to the product backlog.
9. Working on a shared vision, gathering requirements, Managing and prioritizing the Product Backlog, Accepting the software at the end of each iteration, Managing the release plan, The profitability of the project (ROI).
10. The Product Owner achieves initial and ongoing **funding** for the project by creating the project's initial overall requirements, return on investment (ROI) objectives, and release plans.

Scrum Master Responsibility

1. Helping the team understand what goes on during the Daily Scrum
2. Giving the Product Owner guidance in effectively managing the Product Backlog.
3. Giving the rest of the organization guidance on understanding
4. Coaches the team on removing the impediments.
5. Insure the adheres to the practices and rules
6. The Scrum Master is a facilitator, coach, mentor and bulldozer! A Scrum Master is often compared to a sheepdog, responsible for keeping the flock together and the wolves away. Chicken refers to those outside the three Scrum roles.
7. Who is required to attend the Retrospective Meeting? It is attended only by the Team, the Scrum Master, and the Product Owner. The Product Owner is optional.
8. The work of the Project Coordinator is similar to that of the Scrum Master.
9. Agile coaches understand that motivation is not necessarily the problem with performance, but rather capability.
10. In a daily stand-up, the role of the ScrumMaster or team coach is to listen and note any impediments to the team's progress for quick follow-up.

Development Team Responsibility:

1. Team is responsible for managing and putting the ground rules they agreed on
2. Team determines the process to prioritize value in an Agile project
3. Team is estimated should be used for planning
4. Team is responsible for selecting the number of items for the sprint from the product Backlog.
5. The Team is collectively accountable for success or failure of a Scrum project
6. Who determines the process to prioritize value in an Agile project? The team not product owner only.
7. During iteration planning, the product owner's role is to prioritize the backlog items.

The team then decides how many of the top-priority items in the backlog can be completed in the next iteration timebox. So this product owner is overstepping their role, since the amount of work that can be completed in the next iteration is decided by the team, not the product owner or the ScrumMaster.

Scrum events or meeting or ceremonies:

Product backlog refinement (Grooming the Backlog)

1. The product owner owns the backlog
2. Backlog refinement is the prioritization backlog items
3. The entire project team may participate in the backlog grooming

Sprint planning meeting

1. At the beginning of the sprint cycle (every 7–30 days), a “Sprint planning meeting” is held
2. During that meeting, the team first reviews the product backlog, which involves reviewing the overall list of features that will be delivered.
3. The next thing the team should do is create the sprint backlog, which involves extracting items from the product backlog to deliver in the increment for the sprint.
4. Project team needs to discuss the goals of the upcoming sprint and team discusses how the work will be accomplished
5. Development team defines how the work will be done in the goals of the sprint will be achieved
6. The Sprint Planning Meeting is time-boxed to eight hours for a one-month Sprint. For shorter Sprints, the event is proportionately shorter. For example, two-week Sprints have four-hour Sprint Planning Meetings
7. Identify and communicate how much of the work is likely to be done during the current sprint Eight-hour time limit (1st four hours) Entire team: dialog for prioritizing the Product Backlog

(2nd four hours) Development Team: hashing out a plan for the Sprint, resulting in the Sprint Backlog

Daily stand-up (Daily Scrum Meeting)

1. A short meeting, generally conducted every day at the same time and location. All team members are involved in the meeting and stand during the whole meeting.
2. This meeting focuses on mainly three things; what has been done since the previous daily meeting, what is to be done until the next daily meeting and what are the issues or impediments in the accomplishment of tasks.
3. This meeting has no or little significance for the project sponsors and typically not attended by project sponsors
4. This is a 15-minute timeboxed meeting

Sprint review meeting

1. The sprint review meeting is a product-oriented meeting held at the end of every sprint to demonstrate product components completed within that iteration to the product owner, customer and other interested stakeholders.
2. A Sprint Review is held at the end of the Sprint to **inspect** the Increment and **adapt** the Product Backlog if needed. During the Sprint Review, the Scrum Team and stakeholders collaborate about what was **done** in the Sprint. Based on that and any changes to the Product Backlog during the Sprint, attendees collaborate on the next things that could be done. This is an informal meeting, and the presentation of the Increment is intended to elicit feedback and foster collaboration.
3. Review the work that was completed and not completed
4. Present the completed work to the stakeholders (a.k.a. “the demo”)
5. Incomplete work cannot be demonstrated (Team demonstrates completed work)
6. The development team and the product owner will discuss the sprint and the remaining items in the product backlog
7. Four-hour time limit

Retrospective meeting

1. The development team meeting posted after the sprint review, but before the next sprint planning meeting
2. The team check on the effectiveness of the quality process. They look for the root cause of issues
3. Team should continue working on the agreed items till the end of the sprint without changes in the agree backlog. In the retrospective team can check out why they were not able to finish all tasks.
4. Return on Time Invested (ROTI) is used to measure the effectiveness of the retrospective meetings from the team members' perspective.
5. Kaizen is strongly aligned with Agile in the way it encourages changes from the workers rather than asking management what should be changed. Therefore, it's strongly aligned with Retrospective Meeting in Scrum.
6. the benefits of holding regular retrospectives:to **improve capacity**, **productivity**, and **quality**. They do not

directly aim to improve priorities

7. How is the Retrospective meeting BEST conducted?
   The correct sequence is Set the stage, Gather data, Generate Insights, Decide on what to do, Close the Retrospective
8. The iteration retrospective seeks to learn what has gone well, what has not gone well, and what improvements can be made for the next iteration
9. Review of the product owner's feedback about the last iteration, discuss Lessons learned and opportunities for improvement

Scrum Artifacts

Product Backlog

1. The product backlog is an ordered list of "requirements" that is maintained for a product. It contains Product Backlog Items that are ordered by the Product Owner based on considerations like risk, business value, dependencies, date needed,
2. The product backlog is the "What" that will be built, sorted in the relative order it should be built in. It is open and editable by anyone, but the Product Owner is ultimately responsible for ordering the stories on the backlog for the Development Team.
3. The product backlog contains rough estimates of both business value and development effort, these values are often stated in story points using a rounded Fibonacci sequence. Those estimates help the Product Owner to gauge the timeline and may influence ordering of backlog items. For example, if the "add spellcheck" and "add table support" features have the same business value, the one with the smallest development effort will probably have higher priority, because the ROI (Return on Investment) is higher.
4. The Product Backlog, and business value of each listed item is the responsibility of the Product Owner. The estimated effort to complete each backlog item is, however, determined by the Development Team. The team contributes by estimating Items and User-Stories, either in Story-points or in estimated hours.

Sprint Backlog

1. The sprint backlog is the list of work the Development Team must address during the next sprint. The list is derived by selecting stories/features from the top of the product backlog until the Development Team feels it has enough work to fill the sprint.
2. This is done by the Development Team asking "Can we also do this?" and adding stories/features to the sprint backlog. The Development Team should keep in mind the velocity of its previous Sprints (total story points completed from each of the last sprints stories) when selecting stories/features for the new sprint, and use this number as a guide line of how much "effort" they can complete.
3. The stories/features are broken down into tasks by the Development Team, which, as a best practice, should normally be between four and sixteen hours of work. With this level of detail, the Development Team understands exactly what to do, and potentially, anyone can pick a task from the list. Tasks on the sprint backlog are never assigned; rather, tasks are signed up for by the team members as needed during the daily scrum, according to the set priority and the Development Team member skills. This promotes self-organization of the Development Team, and developer buy-in.
4. The sprint backlog is the property of the Development Team, and all included estimates are provided by the Development Team. Often an accompanying task board is used to see and change the state of the tasks of the current sprint, like "to do", "in progress" and "done".

Product Increment

1. The product increment is the outcome of an iteration
2. The product increment is a chunk of the project work
3. The development team and the product owner must be an agreement of what done means for an increment

Very Important Notes on each Domain

Domain I. Agile Principle and mindset Notes:

1. The goal of **agile leadership** is to create an empowered and motivating work environment in which people want to do what needs to be done.
2. **Kanban teams** use their **WIP limits** to control the number of items that can be in a given stage of work at a time.
3. A phased and gated approach is associated with Waterfall. Phases are strictly linear sequences of activities to build a product or deliver a project.
4. Planning and estimating during a feasibility stage is focused on determining value to the customers by asking questions like, "What should we build?" As the project progresses past the feasibility stage, the cost, the budget, and the overall project schedule become clearer and easier to control.
5. **Continuous integration** Small batches of the source code are checked in, the control system initiates a full build, a suite of automated is run, and developer is notified electronically of any code compilation failures
6. **'Barely sufficient'** is the agile term used to describe the approach of doing only as much as is necessary to accomplish the work without including **non-value-added activities**.
7. The scrum value of commitment states 'Because we have great control over our own destiny, we are more committed to success'
8. **Wireframes** assist the project's development by laying out content and functionality so the project has a visual structure and basis for development.
9. The rule of engagement=ground rules = working agreements, provide structure and boundaries for how team members.
10. The Scrum master is responsible for teaching others how to use the Scrum process to deal with every new complexity encountered during a project.
11. Agile approach is an umbrella term that includes a diverse number of Agile methods, practices, and techniques.
12. As uncertainty and risk for a project increase, teams select project delivery life cycles ranging from **Predictive** (non-Agile) approaches to **Agile** approaches.
13. Predictive life cycles take advantage of high certainty around requirements, a stable team, and low overall project risk. These projects are executed in a serial manner.
14. Incremental life cycles optimize work to deliver value to customers more often than a single, final product.
15. Agile techniques and approaches effectively manage **disruptive technologies**.
16. The first principle of agile places customer satisfaction as the highest priority and is key in delivering products and services that delight customers
17. In order to stay competitive, organizations can no longer be internally focused but rather need to focus outwardly to the customer experience.
18. High-uncertainty projects have high rates of change, complexity, and risk.
19. Iterative life cycles are optimized for speed of delivery. They focus on frequent delivery of smaller deliverables.
20. Agile and Kanban are subsets of Lean. This is because they are named instances of lean thinking that share lean concepts as:" focus on value, small batch sizes, & elimination of waste".
21. **Agile manifesto values**
    a. Individuals and interactions over processes and tools
    b. Working software over comprehensive documentation

c. Customer collaboration over contract negotiation
d. Responding to change over following a plan

Domain II. value driven delivery Notes:

1. **Relative prioritization** helps all the stakeholders understand the priority of the features by showing the relative ranking of each user story or work item in relation to all the others.
2. In the **MoSCoW prioritization** scheme, the highest-priority, most fundamental requirements are “must haves”. The next category, “should have” refers to requirements that enable the system to work properly; if we omit those features, the workarounds will be costly or cumbersome. This is the correct answer. The third category “could have”, covers any useful additions that add tangible value to the system but are not required. The remaining option listed here is a distractor.
3. **Increment delivery** is an efficient way to deliver value earlier, reduce rework, and find issues sooner. Although agile teams use it to deliver the customer’s top-priority features first, it isn’t a prioritization method.
4. **A cumulative flow diagram** that plots the flow of the work by individual activities can help us identify the source of bottlenecks on a project.
5. When examining a **cumulative flow diagram** for **bottlenecks**, we look for a widening band for an activity, which indicates the growing completion of work. A widening band is created when one line is followed by another line of shallower gradient.
6. **Continuous integration** is the practice of regularly incorporating new code into the code base and checking to make sure the system still works properly. This helps reveal whether there are any problems in the code so they can be addressed immediately.
7. **The test-driven development** process of writing a test that initially fails, adding code until the test passes, and then refactoring the code is known as either “Red, Green, Refactor” or “red, Green, Clean”.
8. S-curve graphs are easy to interpret and quickly show whether the project is over or under budget by displaying both the estimated and actual spending lines.
9. The main objective of **ruthless testing** is to ensure that product quality remains high throughout the development process.
10. **The requirement Prioritization Model** is a prioritization scheme that involves rating the **benefit, penalty, cost, and risk** associated with each of the features and using a weighted formula to prioritize these features. This method relies on expert judgement rather than on using questionnaires.
11. Defining the product vision, breaking it down into features, and prioritizing them according to value, is
    referred to as ‘Value based analysis and decomposition’.
12. Risk, cost, value are all factors contributing to the decision of how items in the product backlog should be prioritized.

Domain III Stakeholders Engagement Notes:

1. **A decision spectrum** is a tool that a team can use to help them make tough decisions and reach convergence, or consensus.
2. An agile charter would typically be shorter, less detailed, and more focused on how the project will be run (the what, why, when, who, where, and how of the project) than exactly what will be built.
3. **Fist-of-five voting** is a participatory decision-making model in which participants show their level of agreement by voting with their fingers.
4. The collaboration game **Prune the Product Tree** involves drawing a tree on a flip chart and adding the features as leaves at various points on the trunk and branches. This exercise can

help a team brainstorm how the features relate to each other and to the existing functionality. [The goal of 'Pruning the Product Tree' collaboration game is **to gather and shape product requirement**.

5. Brainstorming is useful for many aspects of an agile project, including solving problems, defining the product features, and identifying potential risks.
6. **Agile models** are **lightweight sketches or diagrams** that can be used to present a proposed design, identify its issues and risks, and reach consensus about what will be built. They aren't useful for confirming that the final product works as intended.
7. **Wireframes** are a type of lightweight model, typically a series of sketches or diagrams, that help the team share with their stakeholders how the product they are proposing to build will function, in order to generate discussion and reach consensus.
8. The collaborative game **Remember the Future** involves imagining that the project (or a part of the project, such as a release) has been completed successfully.

Domain IV. Team Performance Notes

1. **Velocity** is tracked in the same metric that the team used to estimate the work--and agile teams can use any name they want for that metric, since it is a relative, rather than an absolute unit of measure. So although "teddy bears" sounds like an unusual way to estimate work, it would be perfectly appropriate from an agile standpoint.
2. **Constructive disagreement** is an important characteristic for a high-performing team. (The other characteristics to look for are self-organizing, empowered, consensus-driven, constructive disagreement, and trust-based.) High-performing teams aren't plan-driven or value-based--we could say that a team's plans and priorities are value-based, but not its people. Team diversity may present both challenges and opportunities, but it isn't a requirement for high performance.
3. To estimate when the work will be finished (at the current rate of progress) on a burndown chart, we extend the average downward slope of the progress line to see when it reaches zero
4. Agile PMOs (Project Management Offices) are multi-disciplinary and invitation oriented. PMOs should not be mandating Agile practices but should be tailoring their engagement based on the needs of the teams applying Agile techniques.
5. Organizational considerations for Project Agility include assessing the organizational culture and structure, procurement management processes business practices and the project management office
6. Collaboration – everything about Agile should encourage collaboration, including communication, co- location, self-organization, decision making, etc., **collaboration is integral to Agile projects**
7. Negotiation – negotiation is an important activity for Agile which yields productive results; plan, schedule, features, changes, etc. are negotiable
8. Active Listening – active listening includes: listening, understanding, retaining and actively responding, can reduce conflicts within the team
9. Conflict Resolution – conflict should be resolved by searching for a "win-win" solution through active listening and communication
10. Servant Leadership – Leaders for Agile projects should practice servant leadership; they are not in the command and control position but rather be a servant to enable the team to perform well and model the right behavior
11. **Adaptive Leadership** – the leader adapts to the environment to lead effectively, i.e. to focus on value- adding activities.

Domain V Adaptive Planning Notes:

1. **Spikes** can be performed at any time during the project, whenever the team wants to explore a risk or problem. **Spikes** are helpful for **learning purposes** and are generally used in circumstances like **acceptance criteria definition**, **estimating and understanding the flow**, and applicability of ideas or actions. Generally, in the cases, **new technology should be investigated** with the help of spikes that makes the team capable of identifying the possible **issues** at earlier stages which otherwise could consume the significant time of the team and lead to project failure
2. **Wideband Delphi** is a potential alternative to using **planning poker**. Both of these approaches allow us to create final estimates iteratively, counter bias, and reach convergence.
3. **Wideband Delphi Estimating** includes **plotting estimates** on a chart with **no names**, and then the range of points is discussed, and the team attempts to reach a consensus.
4. Planning poker is a **fast**, **efficient** way to get reliable estimates for iteration planning.
5. The advantage of using coarse-grained requirements is that it allows us to focus on the big picture and delay detailed product decisions until the last responsible moment.
6. The purpose of performing an **architectural spike** is to perform a "**proof of concept**" check to confirm whether the team's proposed technological approach will work.
7. A "requirements review" = grooming the backlog = refining the backlog.
8. Discrete business-rule tests are the "confirmation" part of the three Cs (the three components of a user story). These tests are explicit statements of the customer's acceptance criteria that show whether a completed feature is working as intended.
9. T-shirt sizing is a high-level estimating technique that is used early in a project to map out the overall effort that will be involved in the project.
10. **Important formula**

- **Schedule Variance (SV)** = Earned value (EV) − Planned value (PV) -- **Cost Variance (CV)** = Earned value (EV) − Actual cost (AC)
- **Schedule Performance Index (SPI)** = (EV) / (PV) **Cost Performance Index (CPI)** = (EV) / (AC)
- **Lead Time** = Work In Progress (units) / Average Completion Rate (units per time period)

Domain VI Problem Detection Notes:

1. **Defect cycle time** is the amount of time bet. when the defect was discovered & when the defect was fixed
2. The **Ishikawa or Fishbone diagram** is one of the commonly used techniques to identify the root cause of a problem and is mostly used in conjunction with the **5 Whys** Technique.
3. **Control limits** are used to detect signals that indicate when a process is not in control, and therefore not operating predictably.
4. **Escaped defects** are the ones not found by the QA team, but by the end users post-release.
5. **WIP** refers to those requirements the team has started working on but are not yet complete. WIP limits are set to minimize sunk cost and waste.
6. **Leading trends** allow us to detect problems before they occur or when they are just starting to happen.
7. Waiting for the next deadline before we start working is an example of "**student syndrome**." While this certainly isn't the best way to work, it isn't one of Cockburn's failure modes.
8. We calculate the expected monetary value of the project threats, so we can prioritize them along with the features in our product backlog to create a risk-adjusted backlog.
9. **Leading metrics** help us adapt and re-plan based on emerging issues.
10. **Lagging metrics** provide a view of historical data, such as budget consumed to date. They don't help us identify opportunities for improvement or give us insight into future trends.
11. **Risk Board** is an information radiator used to make the risks of a project transparent to the team

and the stakeholders.

12. Risk Burndown Chart is a simple graphical indicator of the risk trends in the project.
13. Risk Profile Graphs inform stakeholders if the project risks are increasing or decreasing.
14. The risks identified through the risk management lifecycle are captured and tracked in the Risk Log.
15. **Spike is a time-boxed** period designated to **reduce uncertainty** by learning about a feature, technology, or process.

Domain VII Continuous Improvement Notes:

1. **Releases**, unit tests, acceptance tests, and daily scrums are all forms of **agile reviews**.
2. **Retrospectives** help improve capacity, productivity, and quality.
3. Setting the Stage for a Retrospective (Encourage participation, Set the ground rules, define what people want from the retrospective, have people checking in with one or two words & Working agreements for the retrospective)
4. Demonstrations, prototypes, and simulations are used to clarify requirements and uncover the need for new features.
5. **Value stream map** identify delays, waste and constraints
6. **Project Pre-mortem** Aims to find failure points before they happen
7. **Product Feedback Methods** Prototypes, Simulations & Demos
8. **Brainstorming** Quiet writing, Round robin & Free for all
9. **Round Robin**: Everyone in the team is given a chance to provide their views in a Round Robin fashion.
10. **Free for All**: The team can participate without any restriction and provide inputs.
11. **Quiet Writing**: Team members write their ideas quietly and circulate to the facilitator.
12. **Fishbone Analysis** known as a cause and effect diagram & Ishikawa diagram
13. **Kaizen** is a Japanese word, which stands for ‘continuous improvement or change for good.
14. **Shu** – Follow the Rules. It means the teams that are newly implementing Agile must aim to follow the guidelines provided by the methodology without tailoring any process.
15. **Ha** – Branch out. The team has been using the guidelines and now has a good understanding.
16. **Ri** – Find your own approach. It means that the team has gained sufficient mastery and is creating its own practices and guidelines to suit the project dynamics.
17. **Verification** is looking for defects in the product to analyze how it differs from the intended functionality.
18. **Validation** is to repeatedly check with stakeholders whether the product meets their requirements.
19. **Customer quality** delivers value in the short term while technical quality enables continuous delivery of value over time.
20. **Self-assessment** is a process where an individual, an organization, or a team conducts a comprehensive review of themselves to understand the strengths, weaknesses, and opportunities to improve.