- **Classify software and review the selection of software as a creative tool in order to solve a problem in your workplace.**

The software can be classified into different categories based on its functionality and purpose. Here are some common types of software:

1. **System software:** This software is responsible for managing and operating computer hardware, such as operating systems, device drivers, and utilities (Thüm et al. 2014).
2. **Application software:** This software is designed to perform specific tasks or functions, such as word processing, spreadsheet management, and graphic design (Thüm et al. 2014).
3. **Programming software:** This software is used by developers to write and test code, such as integrated development environments (IDEs) and compilers (Thüm et al. 2014)..
4. **Web software** develops and manages web applications, such as web servers, content management systems (CMS), and web browsers (Finifter, 2011).
5. **Gaming software:** This software is designed for entertainment and includes video games, mobile games, and console games (De Prato, 2010).

When selecting software as a creative tool to solve a problem in the workplace, it is essential to consider the specific needs and requirements of the organisation (Paternoster et al. 2014). For instance, here are some guidelines for selecting software as a creative tool:

1. Identify the problem: Identify the specific problem that needs to be solved and determine what software can help address that problem.

2. Evaluate software features: Evaluate different software options' features and capabilities to determine what best suits the organisation's needs.
3. Consider user experience: Consider how easy the software is to use and how it can fit into the organisation's workflow.
4. Check compatibility: Check the compatibility of the software with existing hardware and software in the organisation.
5. Assess cost and support: Assess the cost of the software and determine whether it provides adequate support and training.

For example, suppose a design firm needs a software solution to create 3D models and visualisations for their clients. In that case, they may evaluate options such as Autodesk 3DS Max, SketchUp, and Blender. Before selecting a software solution, they may consider factors such as ease of use, compatibility with their hardware, cost, and support.

The software can be classified into different categories based on its functionality and purpose. When selecting software as a creative tool to solve a problem in the workplace, it is essential to consider the specific needs and requirements of the organisation, evaluate software features, consider user experience, check compatibility, and assess cost and support.

- **Classify software and review the selection of software as a creative tool in order to solve a problem in your workplace.**

Various software development methodologies can be used to collaborate, design, develop, and test software. Each methodology has its own set of processes and practices that can be tailored to suit the specific needs of a project (Sawant et. al 2012). For instance, here are four popular software development methodologies and how they can be applied in the software development lifecycle:

1. **Waterfall Methodology:** this is a method of developing linear and sequential software. This methodology completes each phase of the software development lifecycle (SDLC) before moving on to the next phase. The phases include requirements gathering, design, development, testing, and maintenance. This methodology is suited for projects with well-defined and stable requirements. The main advantage of the waterfall methodology is that it provides a clear and structured process for development (Heriyanti & Ishak 2020).

2. **Agile Methodology:** this is a method of iterative and incremental software development. This methodology develops the software in small, iterative cycles called sprints. Each sprint involves a set of tasks such as planning, designing, coding, testing, and reviewing. The agile methodology is well-suited for projects with changing or undefined requirements. The main advantage of the agile methodology is that it allows for flexibility and rapid feedback (Almeida et al. 2022).

3. **DevOps Methodology:** The DevOps methodology is an approach to software development that emphasises collaboration and communication between development and operations teams. In this methodology, development and operations teams work together to automate the software delivery process, from design to deployment. The DevOps methodology is well-suited for projects that require frequent updates or deployments. The main advantage of

the DevOps methodology is that it enables faster delivery and more reliable software (Patwardhan et al. 2016).

4. **Scrum Methodology:** The Scrum methodology is an agile framework for software development. It involves a set of roles, events, and artefacts that enable teams to collaborate and deliver software iteratively. The roles include the product owner, scrum master, and development team. The events include sprint planning, daily scrum, sprint review, and sprint retrospective. The artefacts include the product backlog, sprint backlog, and increment. The Scrum methodology is well-suited for complex and adaptive projects. The main advantage of the Scrum methodology is that it enables teams to respond to changing requirements and deliver high-quality software quickly (Samarawickrama & Perera, 2017).

In summary, software development methodologies can be tailored to suit the specific needs of a project. The waterfall methodology is a linear and sequential approach, the agile methodology is an iterative and incremental approach, the DevOps methodology emphasises collaboration between development and operations teams, and the Scrum methodology is an agile framework for software development. Each methodology has advantages and can be used to collaborate, design, develop effectively, and test software.

- **Identify correct methodology for software validation and correctness.**

Various software validation and correctness methodologies can be used to ensure that software is of high quality and meets its requirements. Some of the popular ones are:

1. **Unit testing**: This methodology involves testing each component or module of the software individually to ensure it functions as expected. It helps identify defects early in development and ensures the code is reliable and robust.
2. **Integration testing**: This methodology involves testing the interfaces and interactions between different software components to ensure they work together as expected. It helps detect defects that may arise due to the interactions between components.
3. **System testing:** This methodology involves testing the system to ensure it meets the specified requirements. It helps identify defects that may arise due to integrating different components.
4. **Acceptance testing:** This methodology involves testing the software to ensure it meets the user's requirements and is fit for use. It helps identify defects that may arise due to user expectations and ensures that the software meets the required quality standards.
5. **Automated testing:** This methodology uses tools and scripts to automate the testing process and reduce the required manual effort. It helps detect defects quickly and enables the testing process to be repeated easily.

The methodology choice depends on the software's specific requirements and the stage of the development process. For example, unit testing is more suitable for the early stages of development. In contrast, system and acceptance testing is more suitable for later stages. In addition, a combination of methodologies can be used to ensure that the software is thoroughly tested and validated for correctness.

References:

Thüm, T., Apel, S., Kästner, C., Schaefer, I. & Saake, G. (2014). A classification and survey of analysis strategies for software product lines. *ACM Computing Surveys (CSUR)*, *47*(1), pp.1-45.

De Prato, G., Feijóo, C., Nepelski, D., Bogdanowicz, M. & Simon, J.P. (2010). Born digital/grown digital: Assessing the future competitiveness of the EU video games software industry. *JRC Scientific and Technical Report*, *24555*.

Finifter, M. (2011). Exploring the relationship between web application development tools and security. In *2nd USENIX Conference on Web Application Development (WebApps 11)*.

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T. & Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. *Information and Software Technology*, *56*(10), pp.1200-1218.

Heriyanti, F. & Ishak, A., (2020), May. Design of logistics information system in the finished product warehouse with the waterfall method: review literature. In *IOP Conference Series: Materials Science and Engineering* (Vol. 801, No. 1, p. 012100). IOP Publishing.

Almeida, F., Simões, J. & Lopes, S., 2022. Exploring the benefits of combining DevOps and agile. *Future Internet*, *14*(2), p.63.

Patwardhan, A., Kidd, J., Urena, T. & Rajgopalan, A., (2016). Embracing Agile methodology during DevOps Developer Internship Program. *arXiv preprint arXiv:1607.01893*.

Samarawickrama, S.S. & Perera, I., (2017), September. Continuous scrum: A framework to enhance scrum with DevOps. In *2017 Seventeenth international conference on advances in ICT for emerging regions (ICTer)* (pp. 1-7). IEEE.

Sawant, A.A., Bari, P.H. & Chawan, P.M., (2012). Software testing techniques and strategies. *International Journal of Engineering Research and Applications (IJERA)*, *2*(3), pp.980-986.