P-ISSN: 2986 –4615
Volume 10, No 1, 30 November 2025

DEEPER LEARNING DENGAN *CODING*

Gede Suweken[1], I Wayan Puja Astawa[2], I.N. Sukajaya[3]
[1,2,3]Jurusan Matematika FMIPA UNDIKAHA
Email: gede.suweken@undiksha.ac.id, nyoman.sukajaya@undiksha.ac.id,pja.astawa@undiksha.ac.id

ABSTRACT

The main problem in today's education is the difficulty of engaging students in the learning process, let alone fostering deep learning. For students to achieve deep learning, the first requirement is that they must be actively engaged in learning. Joyful, Mindful, and Meaningful Learning—the pillars of deep learning—are in fact interconnected. Students need to be engaged, ideally with joy, which then allows them to proceed mindfully and meaningfully. As a result of mindful and meaningful learning, students are more likely to experience success in learning, which in turn generates renewed joy. However, the challenge remains: how can we design learning experiences that are joyful, mindful, and meaningful? This is the focus of the present article. The article discusses the outcomes of a Community Service program that trained teachers on how to conduct deep learning. Specifically, it explores the use of coding in education and the rationale for why coding can serve as a vehicle for deep learning. In this program, teachers were trained to program computers—or code—using the visual programming language Scratch or its variants. The goal is for teachers to be able to integrate coding into their teaching practice, thereby enhancing student engagement through such integration.

***Keywords**: engagement, deep learning, visual coding, learning experience.*

ABSTRAK

Masalah utama pembelajaran saat ini adalah sulitnya melibatkan siswa ke dalam pembelajaran. Apalagi membuat siswa belajar secara mendalam (*deep learning*). Agar siswa mencapai pembelajaran yang mendalam, maka persyaratan pertama yang harus dipenuhi adalah bahwa siswa harus terlibat secara aktif dalam pembelajaran. *Joyful, Mindful*, dan *Meaningful Learning*, pilar-pilar *deep learning*, sebenarnya saling berkelindan. Siswa harus terlibat, kalua bisa dengan *joy*, setelah itu baru bisa secara *mindful* dan *meaningful*. Sebagai akibat dari belajar *mindful* dan *meaningful* ini, secara potensial bisa menghasilkan keberhasilan dalam belajar, yang pada gilirannya akan menimbulkan *joy* lagi. Tapi bagaimana menyelenggarakan pembelajaran yang *joyful*, *mindful* dan *meaningful,* inilah yang dibahas dalam artikel ini. Artikel membahas tentang hasil Pengabdian Kepada Masyarakat yang melatih guru-guru tentang cara menyelenggarakan pembelajaran yang mendalam. Artikel membahas tentang pemanfaatan *coding* dalam pembelajaran dan membahas tentang rasional bagaimana *coding* bisa menjadi kendaraan bagi pembelajaran yang mendalam. Pada kegiatan ini, guru-guru akan dilatih memprogram komputer atau *coding* dengan menggunakan bahasa pemrograman visual Scratch atau variannya. Targetnya, guru akan mampu mengintegrasikan *coding* ke dalam pembelajaran, dan dengan pengintegrasian ini keterlibatan siswa dapat ditingkatkan.

***Kata kunci:** keterlibatan, deep learning, coding Scratch, pengalaman belajar.*

PENDAHULUAN

Belakangan ini pendekatan pembelajaran *deep learning* merupakan isu yang sedang hangat didiskusikan orang. Namun, apakah kita, terutama guru-guru sebagai ujung tombak pembelajaran di kelas memang sudah memiliki pemahaman yang betul tentang *deep learning* ini? Video pembelajaran yang dikumpulkan oleh para peserta PPG secara umum berisi antara 3 atau 4 *ice-breaking*, apakah ini dalam rangka impelemntasi *deep learning,* atau secara lebih spesifik *joyful leaning*? Apakah *joyful learning* hanya sebatas nyanyi-nyanyi, nari-nari, atau kegiatan menyenangkan yang sifatnya hanya dipermukaan itu saja? Jika tidak, bagaimana

P-ISSN: 2986 –4615
Volume 10, No 1, 30 November 2025

kita seharusnya membat siswa *joyful* dalam pembelajaran, disamping juga *mindful* dan *meaningful*?

Jika ditelusuri secara mendalam, konsep *deep learning* itu sebenarnya bukan konsep baru. *Meaningful learning*, diperkenalkan oleh David Ausubel sejak tahun 1962. Begitu juga halnya dengan *Mindful learning* (Elen Langer, 1989) dan *joyful learning* (Dewey, early 20th century).

Fokus dari *deep learning* sebenarnya pada pemahaman yang lebih mendalam dan bermakna. *Deep learning* ini membedakan apa yang disebut belajar dipermukaan (*surface learning*) dan belajar secara lebih mendalam (*deeper learning*). Pada *surface learning*, siswa hanya menghapal fakta dan informasi tanpa memahaminya, sedangkan dalam *deeper learning*, siswa memahamai konsep secara mendalam, dalam artian mampu membuat koneksi dengan konsep lain, serta mampu menerapkan konsep dalam berbagai situasi di kehidupan nyata. Pendidikan masa depan harus fokus pada *deeper learning*, untuk menyiapkan generasi yang mampu berpikir kritis, keatif, dan menjadi *problem solver*. Kemampuan ini sangat diperlukan dalam menghadapi kehidupan masa depan yang kompleks, dinamis, dan serba tak terprediksi. Namun masalahnya bagaimana kita menterjadikan pendidikan yang mampu memenuhi tuntutan-tuntutan ini?

Salah satu cara potensial untuk membuat siswa belajar lebih mendalam (*deeper learning*) adalah dengan mengintegrasi *coding* ke dalam pembelajaran. Mengapa begitu dan bagaimana pelaksanaannya secara nyata, itulah yang akan dibahas dalam tulisan ini. Materi dan linatasan pembelajaran yang disampaikan sudah teruji secara tebatas melalu penelitian dan selanjutya disosialisasikan kepada guru-guru melalui kegiatan P2M.

MENGAPA CODING?

Mengapa *coding* harus dibelajarkan kepada siswa? Steve Job pernah mengatakan bahwa "*Everyone in this country should learn how to program a computer ... because it teaches you how to think*." (youtu.be/nKIu9yen5nc). Namun sayang, proses berpikir mungkin sudah lama hilang dari pembelajaran kita, terutama matematika. Guru sering kehabisan cara untuk melibatkan siswa secara aktif dalam memahami konsep matematika yang dibelajarkan (*deeper learning*). Kesulitan ini disebabkan oleh beberapa karakteristik dari generasi Z yang memenuhi kelas-kelas kita saat ini (Prensky, 2009), diantaranya: (i) tidak suka dikuliahi, (ii) ingin dihormati, dipercayai, dan diperhatikan, (iii) ingin mengikuti *passion* mereka sendiri, (iv) ingin berkresi dengan *tools* yang akrab bagi mereka, (v) ingin pendidikan yang tidak hanya relevan tetapi juga real. Jika dikaitkan dengan konsep *deep learning*, maka pembelajaran kita saat ini pada umumnya belum memenuhi konsep *deep learning* yang diharapkan, yaitu
ada dalam pembelajaran, yaitu: *mindful, meaningful*, dan *joyful larning*. Tentu tidak *joy*, karena kebanyakan praktek pembelajaran masih bersifat ceramah dengan peran guru yang dominan, materinya tidak menuntut kreativitas, alat yang digunakan belum sepenuhnya sesuai, dan tidak berguna. Juga tidak *mindful* dan *meaningful*, karena tidak real, tidak sesuai *passion* siswa, tidak terkoneksi secara jelas dengan konsep-konsep sebelumnya.

Lalu, bagaimana *coding* bisa menjadi alat untuk mengimplementasikan *deep*

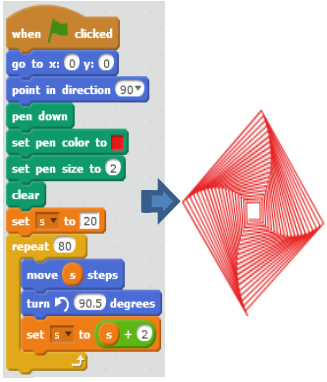
P-ISSN: 2986 –4615
Volume 10, No 1, 30 November 2025

learning? Beberapa kekuatan pembelajaran *coding* yang bisa mendukung *deep learning*, diantaranya

- It uses tools relevant to students' era,
- It teaches perseverance,
- It teaches students how to think and reason,
- It is about creativity and expression,
- It is another way to demonstrate content knowledge,
- It is a way to see math in action. (Heidi William, 2017)

Ha-hal yang disampaikan di atas sangat sesuai dengan konsep *deep learning*.

Berdasarkan pada pengalaman masa lalu, beberapa orang mungkin masih kawatir bahwa *coding* itu sulit dilakukan dan akan mengganggu kelancaran alur pembelajaran. Itu tidak terjadi lagi saat ini. Dengan bahasa pemrograman visual, sebagai alternatif dari bahasa pemrograman tekstual, memprogram menjadi tidak jauh berbeda dengan bermain puzzle. Dalam pemrograman secara visual, seperti misalnya Scratch, siswa hanya perlu fokus pada pemecahan masalahnya, pada penalaran, logika dan urutan perintah (*algoritma*). Sintaks pemrograman tidak akan mejadi beban lagi buat mereka, karena *program*/*coding* dalam pemrograman ini dilakukan hanya dengan mengaitkan keping-keping perintah secara logis. Gambar 1 adalah contoh, bagaimana Scratch digunakan untuk membuat spiral persegi.



Gambar 1: Spiral Persegi.

COMPUTATIONAL THINKING

Dapat dilihat pada Gambar 1 di atas, bahwa aktifitas pemrograman dilakukan dengan cara mengaitkan keping-keping perintah secara logis dan terurut. Jika keping-keping perintah diurut secara logis, maka eksekusi urutan tersebut akan menghasilkan *sesuatu*, yang jika sesuai harapan berarti programnya sudah benar, namun jika hasilnya tidak sesuai harapan berarti masih ada kesalahan logika. Hampir tidak terjadi kesalahan sintaks dalam pemrograman visual ini. Bermain *puzzle* tentu menarik bagi siswa SD, dan jika permainannya bisa dikemas dengan baik dan mengintegrasikan materi pembelajaran, tentu integrasi *coding* ke dalam pembelajaran merupakan hal yang inovatif dan layak untuk dicoba. Tugas selanjutnya adalah merancang tugas atau masalah yang menarik dan menantang dengan tingkat kesulitan yang sesuai dengan ZPD siswa. Apa sebenarnya pemrograman/ *coding* itu? *Coding* sebenarnya perintah kepada komputer, robot, IoT, manusia, dan lain-lain *information processing agents*, yang berisi langkah-langkah logis dan teurut dalam menyelesaikan suatu masalah. *Coding* juga bisa dipandang sebagai realisasi dari *Computational Thinking (CT)* yang tidak lain adalah *problem solving* plus *coding*. *CT* akan membelajarkan siswa tentang:

- Logically organizing and analyzing data,
- Representing data through abstractions such as models and simulations,
- Automating solutions through algorithmic thinking (a series of ordered steps),
- Identifying, analyzing, and implementing possible solutions, with the goal of achieving the most efficient and effective combination of steps and resources,
- Generalizing and transferring this problem-solving process to a wide variety of problems, and

P-ISSN: 2986 –4615
Volume 10, No 1, 30 November 2025

- Formulating problems in a way that enables us to use a computer and other tools to help solve them. (ISTE & CSTA, 2011)

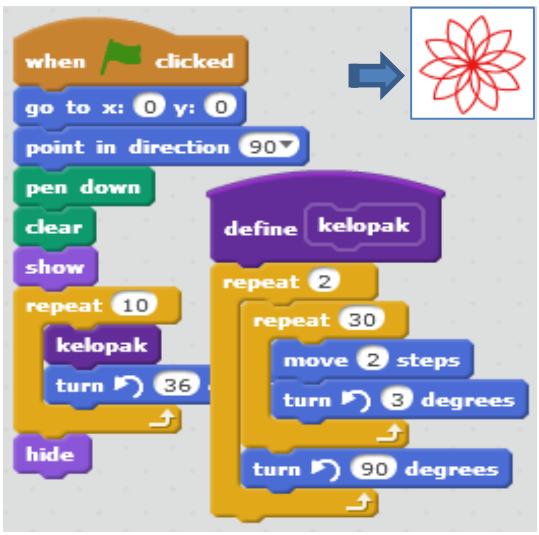
Cikal bakal *CT* adalah Seymour Papert, yang pada tahun 1991, mengagas sebuah paradigm pembelajaran Konstruksionisme (*Costructionism)*. Berbeda dengan konstruktivisme model Piaget, konstruksionisme tidak berhenti hanya pada pentingnya siswa mengkonstruksi konsep, tetapi juga pada pentingnya mengkonstruksi produk yang bermakna bagi dirinya atau orang lain dari konsep yang sudah dikuasai tersebut. Menurut aliran ini, belajar menjadi "*learning by making.*" Papert tertarik pada bagaimana siswa terlibat pada sebuah komunikasi dengan artifak (baik itu buatan sendiri atau orang lain), dan bagaimana komunikasi ini meningkatkan *self-directed* belajar siswa, yang pada gilirannya memfasilitasi konstruksi sebuah pengetahuan yang baru.

Jika dibandingkan dengan instruksionisme, konstruksionisme fokus pada cara membelajarkan siswa agar mampu mengkonstruksi sesuatu yang real, sementara instruksionisme memusatkan perhatian pada cara mengajarkan sesuatu. Keduanya penting, tetapi peningkatan kualitas pembelajaran akan terjadi melalui implementasi paradigm konstruksionsme.

Pada tahun 2006, ide *CT* direvitalisasi oleh Jeanette Wing, seorang ilmuwan komputer. Menurutnya "*Computational thinking involves solving problems, designing systems, and understanding human behaviour, by drawing on the concepts fundamental to computer science*" (Wing, 2006). Menurut Wing, *CT* terdiri dari empat komponen utama, yaitu (i) Dekomposisi, (ii) Abstraksi, (iii) Berpikir Algiritmis, dan (iv) Pengenalan pola.

Dekomposisi adalah proses mengurai masalah yang kompleks menjadi bagian-bagian yang lebih kecil dan lebih mudah dikelola. Bagian yang lebih kecil ini dapat dianalisis secara mandiri, dan dipecahkan, dan sintesa dari solusi masalah-masalah kecil ini diharapkan menjadi solusi dari masalah mula-mula yang kompleks. Sebagai contoh, untuk menggambar grafik fungsi $y = 2x - 6$, misalnya, dekomposisi terdiri dari (i) penentuan titik potong dengan sb. y, (ii) dengan sb. x, dan (iii) penentuan grafik garisnya melalui kedua titik potong tadi.

Abstraksi merupakan bagian dari proses pemecahan masalah yang berkaitan dengan penentuan informasi yang penting dan tidak penting, sehingga dapat dikesampingkan untuk sementara waktu. Misalkan anda belajar mengendarai mobil pertama kali. Maka tak perlu mengingat detail yang tidak perlu, seperti cara kerja mesin, cara memainkan *sound* sistem, atau bahkan cara memakai AC. Cukup ingat hal-hal berikut: (i) Tancapkan kunci kontak, (ii) Hidupkan mobil sambil menekan rem, (iii) kontrol stir sambil menekan gas perlahan, maka mobilakan akan melaju. Dalam *coding,* abstraksi diwujudkan melalui pendefinisian sebuah fungsi atau prosedur. Contoh:



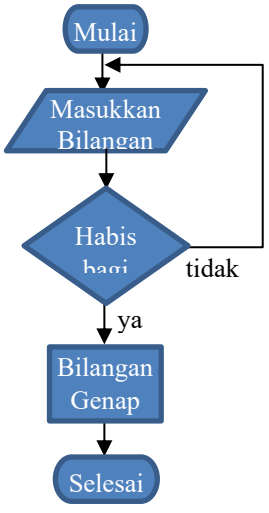
Gambar 2: Penggunaan Blok Baru.

P-ISSN: 2986 –4615
Volume 10, No 1, 30 November 2025

Pada *coding* di atas terlihat pembuatan *bunga 10 kelopak* dengan bantuan prosedur. Dalam hal ini, pembuatan bunga yang kompleks tersebut dilakukan dengan mua-mula menbuat sebuah kelopak saja dalam prosedur (blok) yang disebut "kelopak". Selanjutnya bunga yang besar dilakukan hanya dengan merotasi 36^0 "kelopak" dan kemudian mencetaknya berulang-ulang sebnyak 10 kali. Proses utama menjadi lebih sederhana, karena *detail* pembuatan kelopaknya *dikeluarka*n menjadi prosedur.

Algoritma adalah sekumpulan perintah terurut dan logis yang bisa dilakukan komputer, atau umumnya *information processing agent,* untuk menyelesaikan suatu masalah. Algoritma biasanya dinyatakan dalam bentuk *Flowchart* atau diagram alur. Misalnya, Diagram alur untuk menentukan apakah suatu bilangan genap atau tidak dapat dilihat pada Gambar 3.

Mengidentifikasi pola yang muncul dalam suatu permasalahan bisa sangat membantu dalam memecahkan masalah. Termasuk dalam pengenalan pola misalnya penentuan bentuk-bentuk sama, berbeda atau mudah dimaknai.



Gambar 3: Diagram Alur Penentuan Genap/Tidak

Pola distribusi normal, misalnya, akan cepat dikenali maknanya, bahwa sebagian besar data ada di sekitar rata-rata. Pola pada Gambar 2, dengan mudah bisa dikenali bahwa ia dibentuk dengan sebuah kelopak yang diputar-putar sebanyak 10 kali.

DEEP LEARNING

Dalam pembelajaran, *deep learning* adalah pendekatan yang fokusnya pada pemahaman lebih mendalam, berpikir kritis, dan aplikasi daripada hapalan atau belajar di permukaan (*surface learning*). *Deep learning* menyiapkan siswa untuk mampu mentransfer pengetahuannya, memecahkan masalah-masalah kompleks, serta mengembangkan kompetensi abad 21. Karakteristik dari *deep learning* adalah:

- Fokus pada pemahaman konsep, bukan hapalan,
- Mendorong transfer pengetahuan ke dalam konteks baru,
- Pelibatan proses refleksi dan metakognisi,
- Mendorong kolaborasi dan kreativitas, dan
- Pembelajaran bersifat autentik dan terkoneksi dengan kehidupan nyata.

Di Indonesia dikenal adanya tiga pilar *deep learning*, yaitu (i) *mindful*, (ii) *meaningful*, dan (iii) *joyful* learning. Pilar-pilar ini adalah proses nyata yang harus terjadi agar pembelajaran menjadi *deep learning.*

Siswa harus mengalami dengan sadar bahwa pembelajaran terjadi (*mindful*), bukan dihapal begitu saja, harus ada refleksi, internalisasi, dan metakognisi. Saat belajar Hukum Newton II, misalnya, siswa harus bisa membayangkan bagaimana dalam kenyataan hukum ini bekerja.

Siswa juga harus belajar secara bermakna (*meaningful*). Menurut Ausubel (1963), pembelajaran yang bermakna terjadi dengan cara mengaitkan apa yang sedang dipelajari dengan apa yang sudah

diketahui. Proses asimilasi dan akomodasi diharapkan terjadi dalam hal ini. Jadi siswa tidak cukup hanya menghapal dan mengingat rumus, melainkan ia juga harus mampu mengaitkan rumus tersebut dengan rumus atau konsep sebelumnya. Inilah pentingnya seorang guru harus mampu menstrukturissi materi pelajarannya.

Deep learning sulit terwujud jika siswa tidak *joy*, karena *deep learning* menuntut keterlibatan yang tinggi, terutama keterlibatan kognitif. Namun, *joy* bisa diperoleh tanpa melibatkan hal-hal eksternal yang tidak ada kaitannya dengan pembelajaran. *Wow* atau *eureka effect,* adalah *joy* yang diperoleh tanpa pelibatan hal-hal eksternal; tak perlu ada *ice-breaking*, nyanyi-nyanyi atau nari-nari. Projek pembuatan *game* Scratch, adalah pembelajaran yang potensial menimbulkan *wow effect* (*joy*).

DEEP LEARNING DENGAN CODING

Gambar 2 di atas adalah contoh bagaimana *coding* bisa dijadikan kendaraan untuk *deep learning; mindful, meaningful, dan joyful learning.*

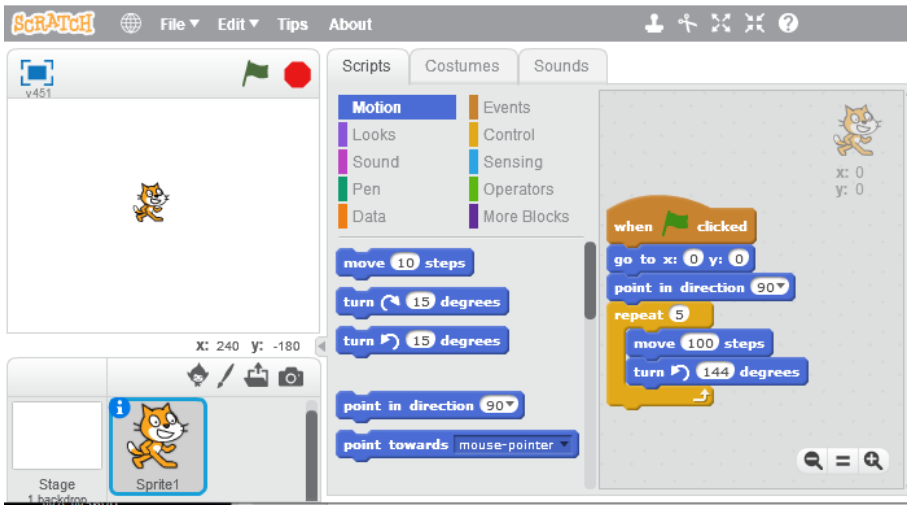
Pertama-tama tampilkan hasil, dan minta siswa melakukan analisis terhadap visualisasi tersebut. *CT* akan merupakan implementasi nyata dari *Mindful learning.* *Bunga 10 kelopak,* sebenarnya adalah rotasi sebuah kelopak sebanyak 10 kali. Buat siswa sadar melakukan dekomposisi.

Meaningful Learning juga terjadi, karena rotasi adalah konsep matematika yang berkaitan dengan kontruksi bunga ini. Jika kita ingin membuat bunga dengan 10 kotak, maka masing-masing rotasinya adalah 36^0, bgaimana jika 12 kelopak? Bagaimana jika rotasinya 45^0, bunganya menjadi berapa kelopak? Ini benar- benar *meaningful learning*.

Joyful learning bisa dipastikan terjadi, karena visualisasinya menarik, tidak terlalu sulit dimaknai, dan potensial menimbulkan *wow effect*. Proses konstruksinya juga *joyful*, karena mirip bermain *puzzle*. Respon sistem yang segera, baik sesuai ataupun tidak dengan harapan dapat menjadi *scaffolding* yang baik bagi siswa, Balikan segera dan *scaffolding* ini akan memotivasi siswa untuk terus mencoba. Bukankah ini esensi dari *joyful learning*?

SCRATCH

Salah satu lingkungan belajar agar terwujud *deep learning* adalah Scratch, sebuah aplikasi pemrograman visual. Scractch merupakan pengembangan lebih lanjut dari



Gambar 4: Scratch 2.0.

Logo yang dikembangkan oleh Papert bersama teamnya. Logo masih bersifat tekstual. Saat ini banyak sekali varian dari Scratch, seperti Snap!, Turbowarp, Turtle Art, BeetleBlock, serta Blockscad, dan lain-lain. Semua aplikasi ini memiliki ciri, bahwa pemrograman dilakukan dengan mengaitkan keeping-keping perintah dalam urutan yang logis. Scratch 2.0 ditampilkan pada Gambar 4.

METODE

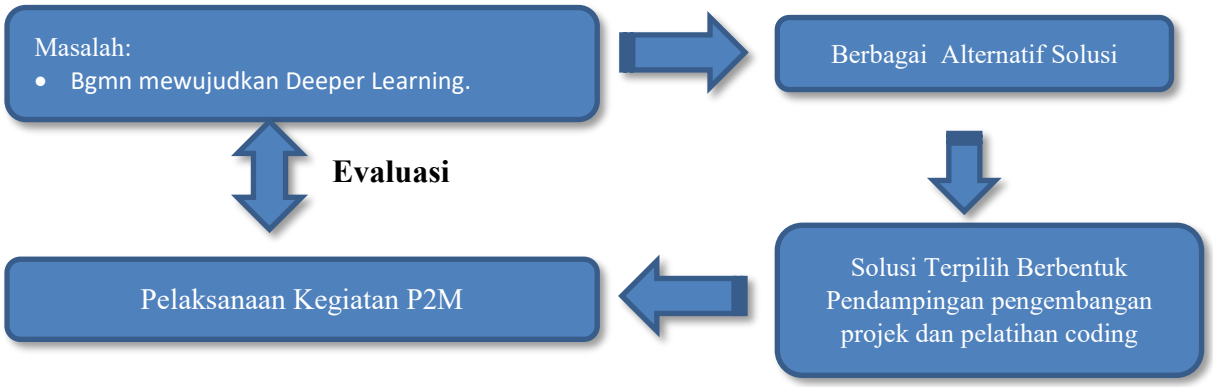
Kegiatan P2M diselenggarakan secara luring di SMPN Satap 1 Kubutambahan Singaraja dengan rincian:

1. Ekspositori tentang cara mendisain atau mendapatkan projek yang berkaitan dengan matematika,
2. Pembelajaran dan pelatihan mengkoding dengan *Scratch* melalui PBL,
3. Latihan dan diskusi pembuatan *Scratch coding* dengan materi kajian Geometrid an Aljabar,
4. Penugasan

HASIL DAN PEMBAHASAN

Kegiatan P2M ini dilakukan secara luring selama 1 hari pada tanggal 9 Agustus 2025. Kegiatan kemudian dilanjutkan dengan tugas terbimbing dengan pembimbingan melalui grup WA. Kegiatan tatap muka diisi dengan teori tentang cara meningkatkan *engagement* dalam pembelajaran terutama bagi siswa yang *digital natives*, *learning by making*, *Computational Thinking* (*CT*), pengenalan Scratch, dan contoh-contoh pembuatan *coding* Scratch. Sementara tugas mandiri diserahkan topiknya kepada masing-masing guru.

Pada hari pertama, berbagai teori tentang karakteristik siswa *digital natives* (Prensky, 2009) disampaikan. Juga dibahas tentang paradigma pembelajaran *learning by making* dari Seymour Papert (Papert dan Harel, 2002). Selanjutnya dibahas tentang bahasa pemrograman (*coding*) berbasis visual yang relatif mudah digunakan dibandingkan dengan bahasa pemrograman berbasis text. Dengan *visual-based coding* siswa tidak lagi dibebani oleh sintaks program melainkan fokus pada logika dan algoritma penyelesian masalah.



Gambar 5: Diagram Kerangka Pemecahan Masalah.

Selanjutnya materi yang disampaikan adalah berbagai perintah Scratch dan pemanfaatannya sebagai alat untuk membuat coding. Secara rinci materi yang dibahas pada bagian ini adalah sebagai berikut:

1. *Computatonal Thinking* (CT),
2. Scratch dalam pengembangan CT
3. Pengenalan Scratch, sprite, latar, block.

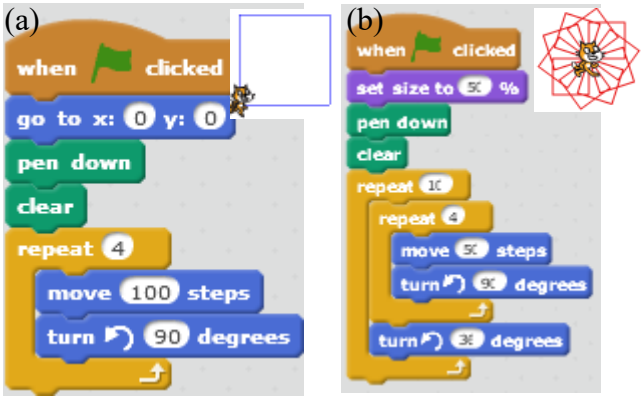
4. Media pembelajaran dengan Scratch,
5. Presentasi.

Berikut adalah beberapa dokumentasi dari pelaksanaan kegiatan P2M ini.

Gambar 6: Pembukaan P2M.

Gambar 7: Praktek Coding Scratch.

Berikut adalah beberapa *coding* yang dihasilkan oleh guru:



Gambar 8: Beberapa Karya Guru & Siswa.

SIMPULAN

Melalui kegiatan P2M ini, guru-guru SMPN Satu Atap 2 Kubutambahan, Singaraja telah mendapatkan pengetahuan tambahan tentang pemrograman komputer dengan menggunakan Bahasa pemrograman visual Scratch. Yang lebih mengesankan, kegiatan ini diikuti bukan hanya oleh guru matematika, melainkan juga oleh beberapa siswa.

DAFTAR RUJUKAN

1. Calao, Luis Alberto, et.al. *Developing Mathematical Thinking with Scratch, An Experiment with 6th Grade Students*. © Springer International Publishing Switzerland 2015, DOI: 10.1007/978-3-319-24258-32.
2. Calder, Nigel. 2010. *Using Scratch: An Integrated Problem Solving Approach to Mathematical Thinking*. APCM 15(4)2010.
3. Computing At School. 2014. *Computing in the national curriculum. A guide for secondary teachers*. UK: NAACE.
4. Papert, Seymour. 1999. *Logo Phylosophy and Implementation*. Logo computer System Inc.
5. Papert, Seymour and Idit Harel. 2002. *Situating Constructionism*. Digital Nations: MIT Media Lab.
6. Prensky, Marc. 2001. *Digital Natives, Digital Immigrant.* (From **On the Horizon (MCB University Press, Vol. 9 No. 5, Oct., 2001).**
5. Prensky, Marc. 2009. *Teaching Digital Natives. Partnering for Real Learning.* UK: Center for Excelence in Media Pratice.
6. Resnick, Mitchel, John Maloney, and Natalie Rusk. 2009. *Scratch: Programming for All*.
7. Communication of the ACM, November 2009. DOI: 10.1145/1592761.1592779
8. Small, Marian. 2017. *Teaching Mathematical Thinking*. New York: Teachers Colleges Press.
9. William, Heidi. 2017. *No Fear Coding. Computational Thinking Across The K5 Curriculum*. USA: ISTE.